NUWC-NPT Technical Document 11,800
2 April 2007

User's Manual for the Tonpilz-Window Interaction Model (TWIM) Program

Andrew J. Hull
Autonomous Systems and Technology Department

NAVSEA
WARFARE CENTERS
NEWPORT

Naval Undersea Warfare Center Division
Newport, Rhode Island

Approved for public release; distribution is unlimited.

PREFACE

The work described in this document was prepared under Project N07PR01253-00, "Broadband Acoustic Window Designs for Heavyweight and Lightweight Torpedoes," principal investigator Andrew J. Hull (Code 8212). The sponsoring activity is the Office of Naval Research, program manager David Drumheller (ONR 332).

Reviewed and Approved: 2 April 2007

James S. Griffin
Head, Autonomous Systems and Technology Department

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY ***(Leave blank)***

2. REPORT DATE
2 April 2007

3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE

User's Manual for the Tonpilz-Window Interaction Model (TWIM) Program

5. FUNDING NUMBERS

6. AUTHOR(S)

Andrew J. Hull

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Undersea Warfare Center Division
1176 Howell Street
Newport, RI 02841-1708

8. PERFORMING ORGANIZATION REPORT NUMBER

TD 11,800

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
875 North Randolph Street
Arlington VA 22203-1995

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The Tonpilz-Window Interaction Model (TWIM) program calculates the element receive response of an array of Tonpilz sensors that resides behind an elastomeric sonar window loaded on the front side with a fluid that contains an acoustic pressure. This program allows the user to build a mathematical model of a Tonpilz array that is in contact with a fluid-loaded sonar window. This user's manual explains the program in detail and includes sample output and flowcharts.

14. SUBJECT TERMS

Sonar Systems   Tonpilz Sensors   Tonpilz-Window Interaction Model (TWIM)

15. NUMBER OF PAGES
56

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Cont'd)

LIST OF TABLES

USER'S MANUAL FOR THE TONPILZ-WINDOW INTERACTION MODEL (TWIM) PROGRAM

1. INTRODUCTION

The Tonpilz-Window Interaction Model (TWIM) is a MATLAB program that calculates the element receive response of an array of Tonpilz sensors that resides behind an elastomeric sonar window loaded on the front side with a fluid that contains an acoustic pressure. This configuration, shown in figure 1, corresponds to many typical sonar systems that are designed to detect incoming acoustic energy. The TWIM program allows the user to enter the parameters of the sonar system based on the size of the array, sensor head and tail mass, sensor stiffness, and material properties of the sonar window. The program then calculates the dynamic response of an element of the array versus frequency and wavenumber or arrival angle. The analysis consists of three basic parts: user configuration input, analytical calculations, and user post-processing. After the analysis, the user can post-process the data to produce output in the desired graphical form.

The TWIM program version number consists of three numbers separated by dots; the current program is version 1.0.7. The first number is the primary program version, the second is the revision of that version, and the third is the version of MATLAB that is compatible with the TWIM program. This program was designed with MATLAB 7.3. The program also comes with a stand-alone routine entitled "MakeDefaultSet." This routine generates the binary file that TWIM reads to start the application and is included in case the binary file is unreadable on the computer in use. Thus, if the program is ported to a different computer and version of MATLAB, it can still be used.

The theoretical calculations for the TWIM program were developed in reference 1 (NUWC-NPT TR 11,781). The governing equations and solutions are not presented in this document, which was written solely to provide documentation of the TWIM program. Program validation is included in this document and is based on several test problems.

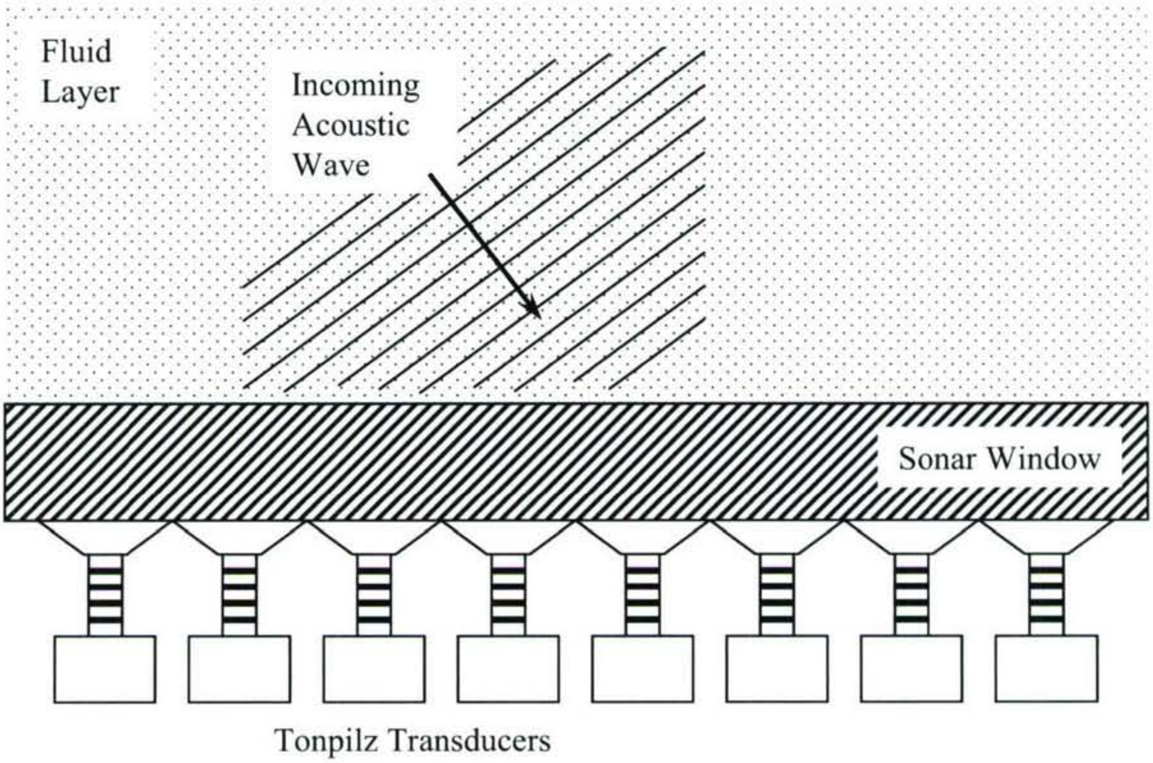

Figure 1. Modeled Configuration for the TWIM Program

FILENAMES AND TYPES

The TWIM program consists of 10 MATLAB subroutines. The filename of each subroutine has the suffix ".m", which identifies it as a MATLAB file. Additionally, there is an 11^{th} file that is used for data management that is a MATLAB binary file and a 12^{th} file that is a stand-alone MATLAB program designed to generate the 11^{th} file in the event that it is unreadable on the computer. The files that compose the TWIM program and a brief description of each are:

Twim.m (Tonpilz-Window Interaction Model program) is the upper-level routine that starts the program and directs the user to select either the input or post-processing modules. The background colors of the graphic boxes and plots are set in this module. Additionally, the text size of the graphical displays can be changed in this routine.

InputControlPanel.m displays the graphical user interface (GUI) input module panel so that the user can view and change the parameters associated with the analysis. The computational analysis is initiated from this control panel.

ParameterControlPanel.m displays separate GUI panels so that the user can change the parameters associated with the analysis. Four different GUI panels can be initiated in this routine: Fluid Properties, Sonar Window Properties, Sensor Parameters, and Analysis Parameters.

EditParameters.m reads the analysis parameters from the individual GUI control panels and updates the memory with the new parameters.

AnalysisModule.m performs the analytical calculations associated with the analysis.

PostProcessData.m provides the GUI control panel that allows the user to post-process the data from an existing analysis or from a previous analysis.

EditPostProData.m reads the user inputs from the post-process GUI and allows the plots to be generated.

DispParameter.m displays the parameters (as text) from the current analysis set that is loaded into the program.

PlotRoutine.m provides the plotting algorithms that allow graphical display of the data.

WritePrevValTwim.m writes the parameter values from the previous analysis to a file named **PreValTwim.mat** so that each new analysis can load the values into the memory and use them as the initial default values.

PreValTwim.mat is the MATLAB binary data file that contains the parameter values from the previous analysis.

Additionally, the program **MakeDefaultSet.m** is included in the TWIM program. This is a MATLAB file that generates the file **PreValTwim.mat** on the current computer in case the file is imported from a computer operating system or version of MATLAB that is incompatible with the computer the user is currently operating. This program does not interact with TWIM and is not required for TWIM to operate.

STARTING THE PROGRAM

The TWIM program is initiated by starting MATLAB and changing the present working directory to the directory in which TWIM resides. Next, *Twim* is typed in the MATLAB command window to start the program. Because all further operations are graphically driven, there is no need to interact with the MATLAB programming language. When the TWIM program is started, the top-level user interactive GUI box will appear with 10 options (figure 2). The first three options are program commands, the next three are graphic display commands, and the final four are screen font size selection commands.

The program analysis options are:

- **Generate Element Receive Pattern of Tonpilz Transducer**, which directs the program to the GUI input module.

- **Graphically Post-Process Previous Data**, which directs the program to the GUI plotting module.

- **Quit TWIM Program**, which quits the program and closes MATLAB.

The program color settings are:

- **Black background on plots with black border**. This command places a black background and border on all plots. Graphic interface boxes will also have a black background.

- **White background on plots with grey border**. This command places a white background and grey border on all plots. Graphic interface boxes will have a grey background.

- **Black background on plots with grey border**. This command places a black background and grey border on all plots. Graphic interface boxes will have a grey background.

The font size selection options are: 8 point, 10 point, 12 point, and 14 point.

When the program is started, the background and border colors that were used in the preceding analysis are the default values. These can be changed immediately by use of the pushbutton command in the first GUI.

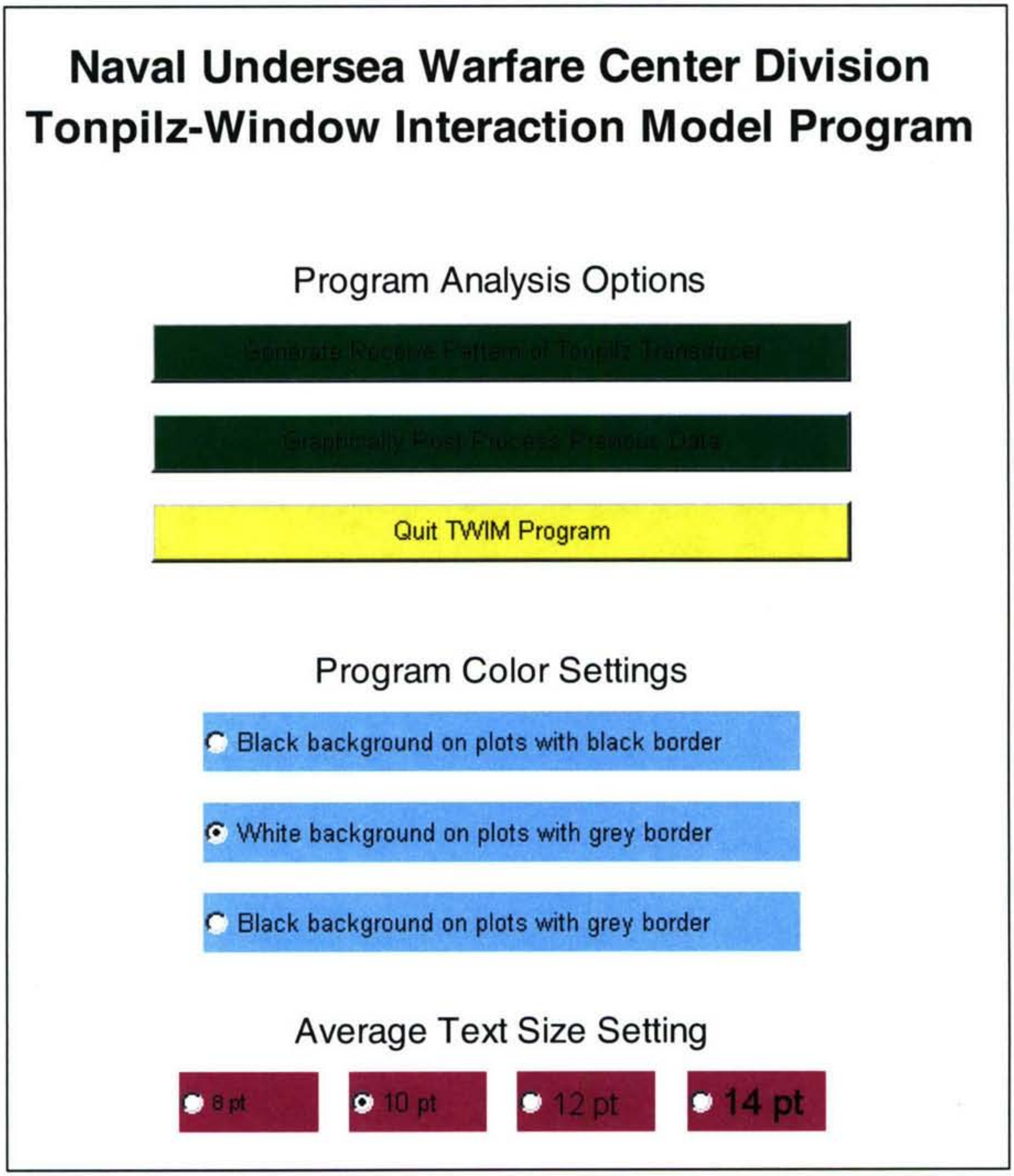

Figure 2. Top Level Interactive Graphical User Interface

INPUT CONTROL PANEL COMMANDS

The input control panel is entered from the initial control panel by pressing the button **Generate Element Receive Pattern of Tonpilz Transducer** (see figure 2). When the input module control panel is opened, the program default values are those that were used in the preceding analysis. The input control panel is used to change the parameters of the analysis. The parameters are grouped into five subcategories depending on their relationship to the analysis. Each parameter that is to be changed has to be done so by pressing the **Change** button next to the parameter. The input control panel GUI is shown in figure 3.

1. **Fluid Properties**

- **Compressional Wave Speed**—allows the user to change the compressional wavespeed of the fluid that is in contact with the elastomeric layer. The units are meters per second (m/s).

- **Density**—allows the user to change the density of the fluid that is in contact with the elastomeric layer. The units are kilograms per cubic meter (kg/m^3).

2. **Sonar Window Properties**

- **Young's Modulus**—allows the user to change Young's modulus of the sonar window. The units are newtons per square meter (N/m^2) or pascals (Pa).

- **Loss Factor**—allows the user to change the loss factor associated with Young's modulus of the sonar window. This is a dimensionless quantity.

- **Shear Modulus**—allows the user to change the shear modulus of the sonar window. The units are newtons per square meter (N/m^2) or pascals (Pa).

- **Loss Factor**—allows the user to change the loss factor associated with the shear modulus of the sonar window. This is a dimensionless quantity.

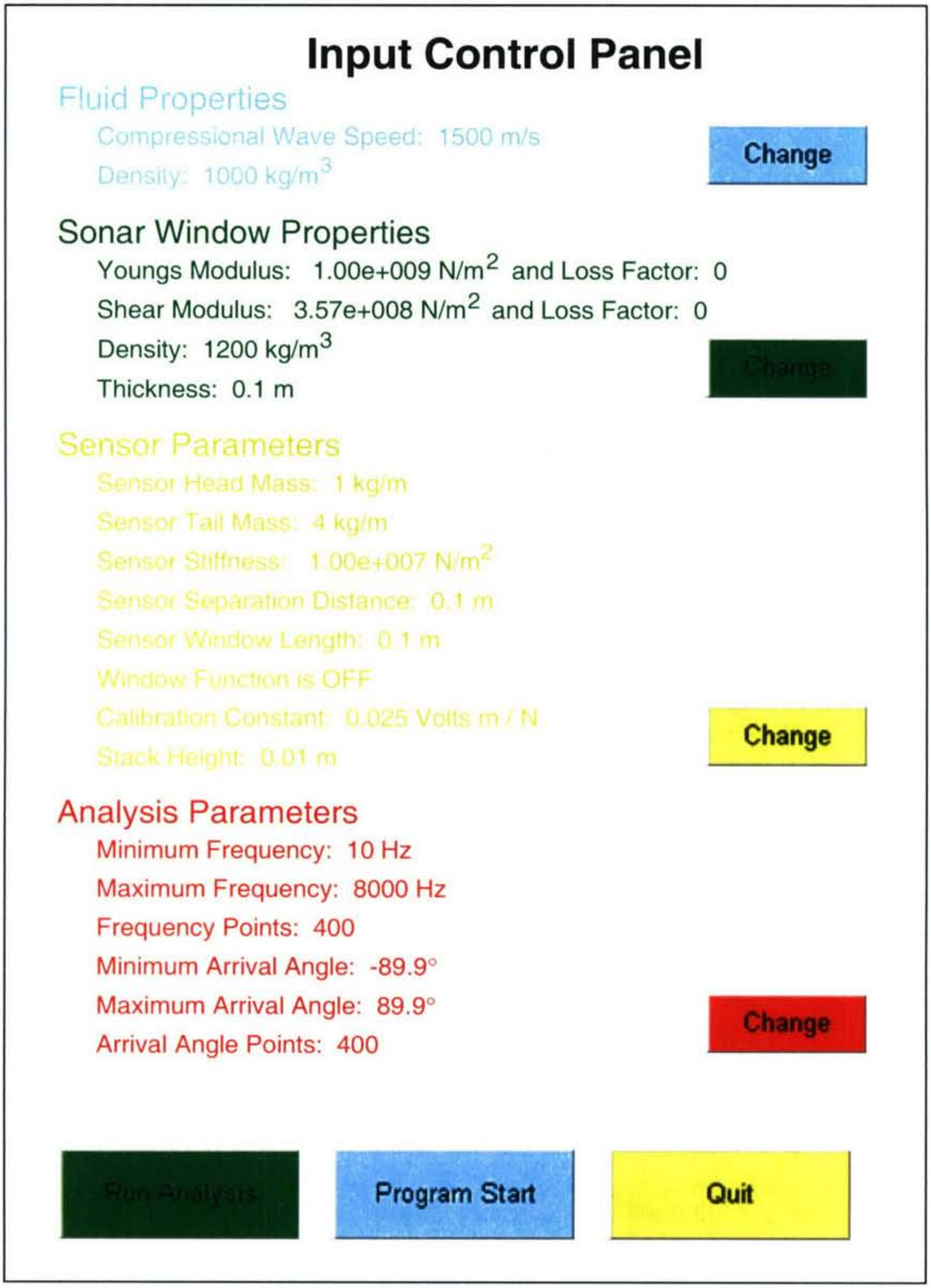

Figure 3. Input Control Panel Graphical User Interface

- **Density**—allows the user to change the density of the sonar window. The units are in kilograms per cubic meter (kg/m^3).

- **Thickness**—allows the user to change the thickness of the sonar window. The units are in meters (m).

3. **Sensor Parameters**

- **Sensor Head Mass**—allows the user to change the Tonpilz head mass. The units are in kilograms per meter (kg/m).

- **Sensor Tail Mass**—allows the user to change the Tonpilz tail mass. The units are in kilograms per meter (kg/m).

- **Sensor Stiffness**—allows the user to change the Tonpilz stiffness. The units are in newtons per square meter (N/m^2).

- **Sensor Separation Distance**—allows the user to change the sensor separation distance; i.e., the distance from one sensor to the adjacent sensor. The units are in meters (m).

- **Sensor Window Length**—allows the user to change the sensor length in the window function. This value is typically very close to the sensor separation distance. The units are in meters (m).

- **Window Function Not Included in Model**—allows the user to turn the window function off during the analysis. It is toggled off by the next button in the window that is brought up by using the yellow change button.

- **Window Function Included in Model**—allows the user to turn the window function on during the analysis. It is toggled off by the previous button in the window that is brought up by using the yellow change button. The window function is a $|[\sin(kL/2)]/(kL/2)|$ window.

- **Calibration Constant**—allows the user to change the Tonpilz piezoelectric calibration constant that relates the sensors stiffness to its voltage output. The units are in volts times meters per newton (volts m/N).

- **Stack Height**—allows the user to change the stack height of the Tonpilz sensor. The units are in meters (m).

4. **Analysis Parameters**

- **Minimum Frequency**—allows the user to change the minimum frequency value of the analysis. The units are in Hertz (Hz).

- **Maximum Frequency**—allows the user to change the maximum frequency value of the analysis. The units are in Hertz (Hz).

- **Number of Frequency Points**—allows the user to change the number of frequency points in the analysis.

- **Output Versus Wavenumber**—allows the user to set the analysis output versus wavenumber. It is set for the next window by pressing the red change button. It is toggled off by the next button.

- **Output Versus Arrival Angle**—allows the user to set the analysis output versus arrival angle. It is set for the next window by pressing the red change button. It is toggled off by the previous button.

- **Minimum Wavenumber or Minimum Arrival Angle**—allows the user to set the minimum wavenumber or minimum arrival angle associated with the analysis. If **Output Versus Waveumber** is active, the screen will read **Minimum Wavenumber**. If **Output Versus Arrival Angle** is active, the screen will read **Minimum Arrival Angle**. The units are in radians per meter (rad/m) for wavenumber and degrees (deg) for arrival angle.

- **Maximum Wavenumber or Maximum Arrival Angle**—allows the user to set the maximum wavenumber or maximum arrival angle associated with the analysis. If **Output Versus Waveumber** is active, the screen will read **Maximum Wavenumber**. If **Output Versus Arrival Angle** is active, the screen will read **Maximum Arrival Angle**. The units are in radians per meter (rad/m) for wavenumber and degrees (deg) for arrival angle.

- **Number of Wavenumber Points or Number of Arrival Points**—allows the user to set the number of wavenumber points or number of arrival angle points, based on whether **Output Versus Waveumber** is active or **Output Versus Arrival Angle** is active.

5. **Analysis Commands**

- **Run Analysis**—initiates the analysis of the system.

- **Program Start**—returns the program to the initial startup screen.

- **Quit**—quits the program and closes MATLAB.

ANALYSIS CONTROL PANEL COMMANDS

The analysis module (figure 4) is where the program computes the dynamic response of the currently defined model. It is entered using the **Run Analysis** button in the input module. The analysis module has no user input. When the analysis is being computed, a green progression bar appears on the screen and is updated every time a single frequency computation is completed. The analysis module has a **Quit Analysis** button that terminates the computations and returns the program to the initial startup module. All current computations are deleted when this button is pressed. Before the analysis begins, the program prompts the user for a filename to store the data. Response to this prompt ensures that the data are written to a file rather than to an optional command, where the data may be lost because the user neglects to save them. This file is written as a MATLAB binary file. The system parameters used in the calculations are also stored with the results. After the analysis data are stored, the program is routed into the post-process module where the current analysis parameters are displayed and the analytical results can be plotted. The MATLAB code of the analysis module is included as an appendix in this document.

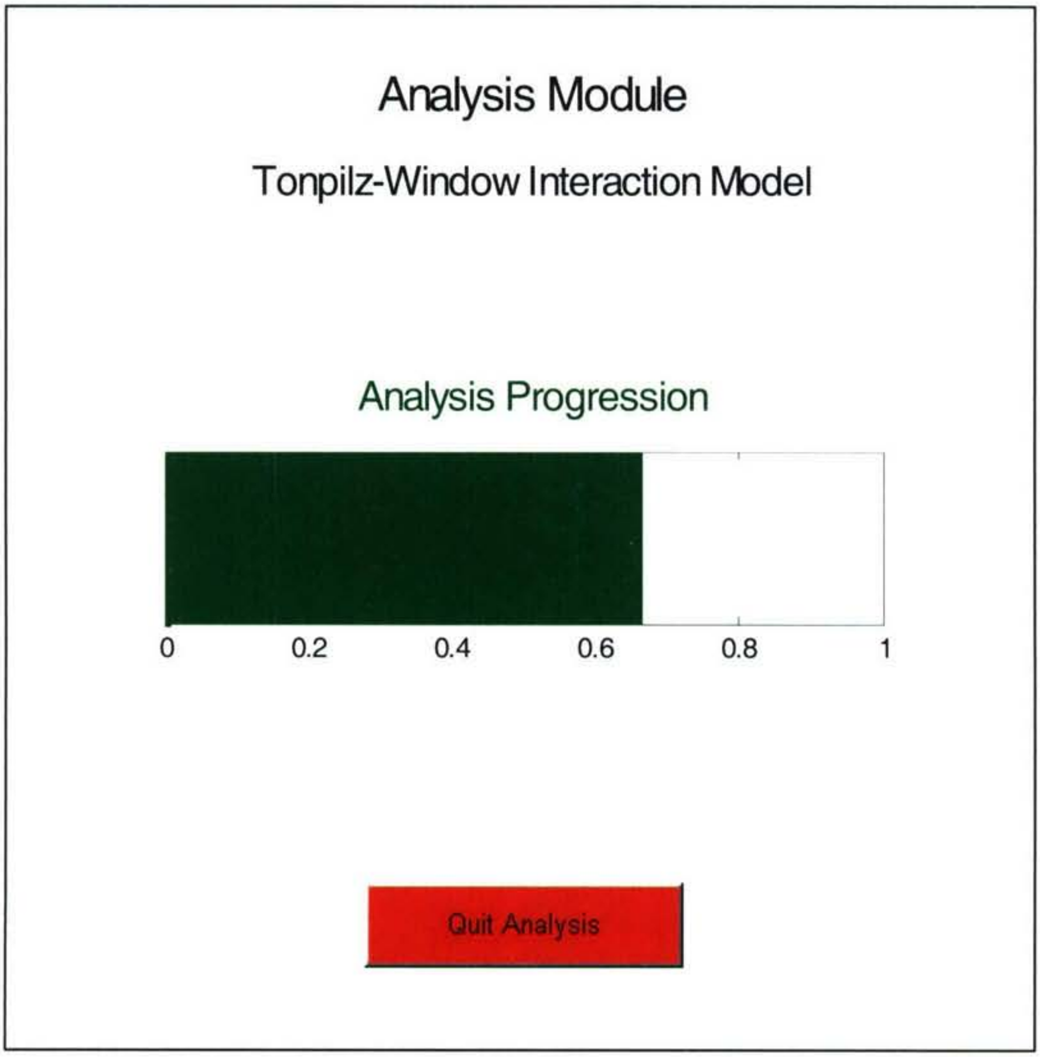

Figure 4. Analysis Module Graphical User Interface

POST-PROCESS DATA PANEL COMMANDS

The post-process module allows the data to be plotted using various options. There are two different ways to enter this module. The first is at initial TWIM startup when the graphical **Post-Process Previous Data** button is selected and the program goes directly to the plotting module. The second way to enter the module occurs after an analysis, when the analysis module produces a calculation. In this mode, the program will enter the post-process module and the just calculated data will be loaded into the post-processing routine. The plotting control panel has a number of options for post-processing data. The post-process GUI is shown in figure 5.

- **Surface Plot of Output Versus Frequency and Wavenumber or Arrival Angle**—allows the user to plot the output as a color surface with frequency as the x-axis and wavenumber or arrival angle as the y-axis. It is toggled off by the next two buttons.

- **Line Plot of Output Versus Wavenumber** or **Arrival Angle at ___Hz**—allows the user to plot a constant frequency cut at an entered frequency value. The program finds the nearest frequency bin to the one requested and plots the data versus wavenumber or arrival angle, depending on which quantity was used to generate the data set. It is toggled off by the preceding and following buttons. The default value is the smallest analysis frequency.

- **Line Plot of Output Versus Frequency at ___ rad/m** or **degrees**—allows the user to plot a constant wavenumber or arrival angle cut, at an entered value, depending on which quantity was used to generate the data set. The program finds the nearest wavenumber or arrival angle bin to the one requested and plots the data versus frequency. It is toggled off by the two preceding buttons. The default value is the smallest analysis wavenumber or arrival angle.

- **Plot Size ___inches by ___inches**—allows the user to define the size of the plot that is to be generated to the screen in units of inches. The first box is the horizontal dimension and the second is the vertical direction. The default size is 6.5 inches by 4.5 inches.

- **Normalization**—offers three options (**None, 0 dB Global, 0 dB Each Bin**) that are used to control the normalization of the plotted data and are toggled on/off with respect to each other. The first option (**None**) does not apply any type of normalization to the data; the second option (**0 dB Global**) makes the maximum value of the data 0 dB; and the third option (**0 dB Each Bin**) makes the maximum value of each frequency bin 0 dB. The default value is **None**.

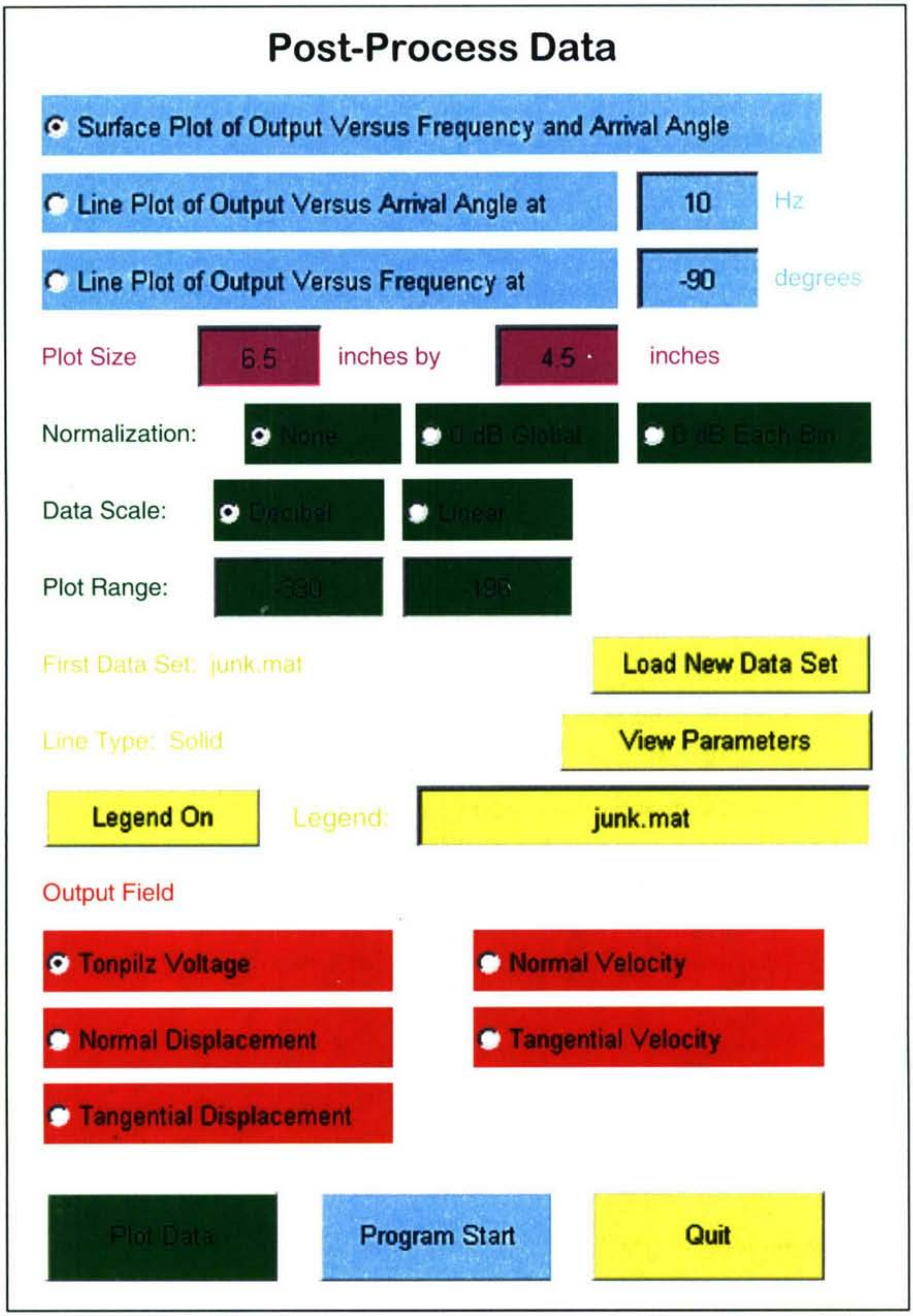

Figure 5. Post-Process Data Graphical User Interface

- **Data Scale**—offers two options (**Decibel** or **Linear**) that are used to set the scale of the displayed data and are toggled on/off with respect to each other. The first option (**Decibel**) sets the display of the data to a decibel scale, and the second option (**Linear**) sets the display to a linear scale. The default value is **Decibel**.

- **Plot Range**—sets the range of the plot. The default values are the minimum and maximum values of the data that are to be plotted. The first value must always be less than the second value.

- **Load New Data Set**—allows the **First Data Set** to be changed by loading an existing analysis file into the program.

- **View Parameters**—displays a text window that contains the parameters associated with the analysis that is loaded in the program.

- **Legend On/Off**—controls the legend on the line plots. It self-toggles on/off. The default value is **On**.

- **Legend**—displays the text as a legend on the line plots. The default text is the filename of the data set.

- **Tonpilz Voltage**—selects the Tonpilz voltage as the data from the analysis that will be displayed in the plots.

- **Normal Displacement**—selects the normal displacement at the bottom of the plate as the data from the analysis that will be displayed in the plots.

- **Tangential Displacement**—selects the tangential displacement at the bottom of the plate as the data from the analysis that will be displayed in the plots.

- **Normal Velocity**—selects the normal velocity at the bottom of the plate as the data from the analysis that will be displayed in the plots.

- **Tangential Velocity**—selects the tangential velocity at the bottom of the plate as the data from the analysis that will be displayed in the plots.

- **Plot Data**—displays the plot of the data based on the post-process commands that are set in the window.

- **Program Start**—returns the program to the initial startup screen.

- **Quit**—ends the program and closes MATLAB.

SAMPLE OUTPUT AND FLOWCHART

Sample output from the program is illustrated in this section. The output corresponds to the parameters listed in table 1, which is also the parameter set in the **MakeDefaultSet.m** file and the numerical example that is given in reference 1.

Table 1. Parameters Used to Establish Default Data Set

Parameters	Value
Density of Fluid	1000 kg/m^3
Compressional Wavespeed of Fluid	1500 m/s
Young's Modulus of Window	1.00e9 N/m^2
Young's Modulus Loss Factor	0.0 (dimensionless)
Shear Modulus of Window	3.57e8 N/m^2
Shear Modulus Loss Factor	0.0 (dimensionless)
Density of Window	1200 kg/m^3
Thickness of Window	0.100 m
Tonpilz Head Mass	1.00 kg/m
Tonpilz Tail Mass	4.00 kg/m
Tonpilz Stiffness	1e7 N/m^2
Separation Distance of Sensor	0.100 m
Transducer Constant	0.025 volts m/N
Sensor Window	Off
Sensor Length	0.100 m
Sensor Stack Width	0.095 m
Sensor Stack Height	0.010 m
Minimum Frequency	10 Hz
Maximum Frequency	8000 Hz
Frequency Points	400
Minimum Arrival Angle	-89°
Maximum Arrival Angle	89°
Number of Arrival Angle Points	400

Figure 6 is a color image of the Tonpilz output voltage divided by input pressure versus frequency and arrival angle. Figure 7 is a line plot of Tonpilz output versus arrival angle at 2493 Hz. In this figure, a constant frequency of 2500 Hz was requested for the plot. The plot is at the nearest frequency bin to the requested frequency. Figure 8 is a line plot of Tonpilz output versus frequency at -0.2°. In this figure, a constant arrival angle of 0° was requested and the nearest arrival angle bin was -0.2°. A flowchart of the program is shown in figure 9.

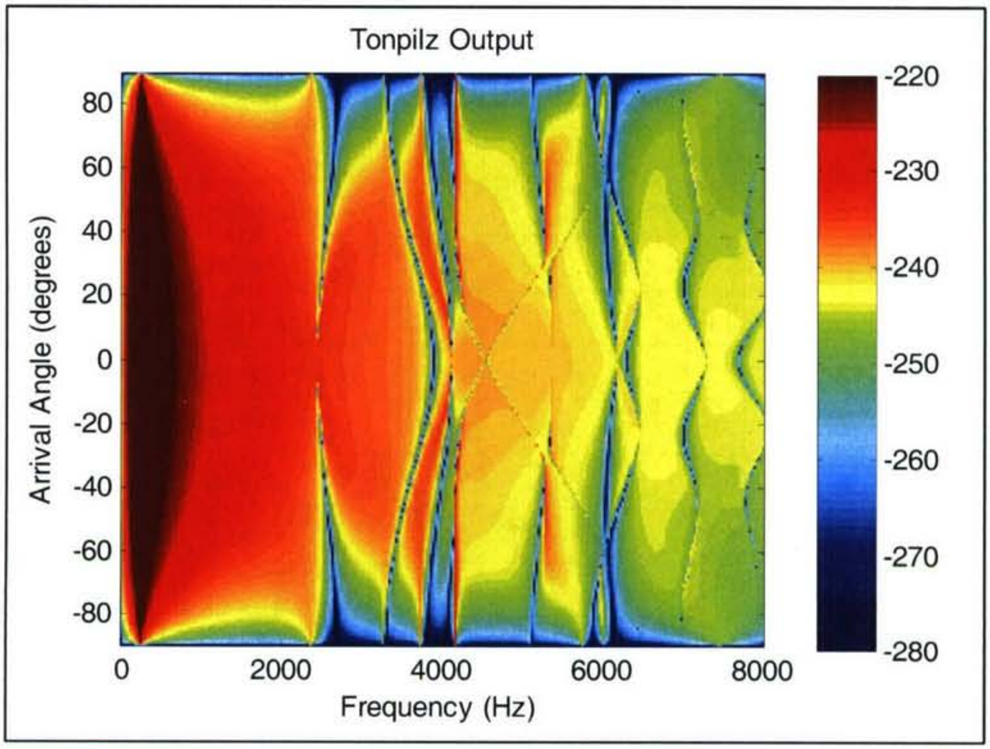

Figure 6. Tonpilz Output Versus Frequency and Arrival Angle

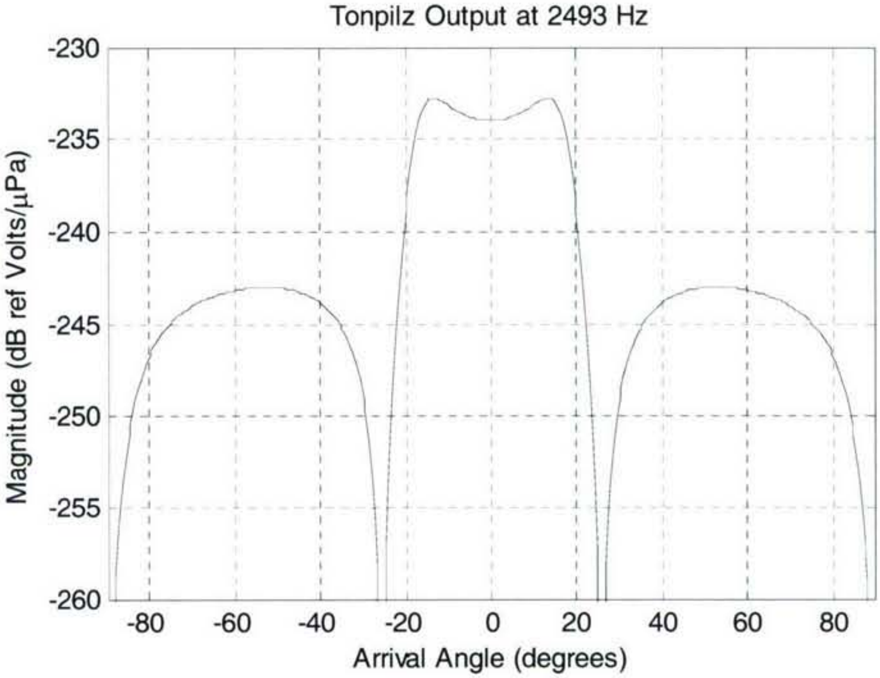

Figure 7. Tonpilz Output Versus Arrival Angle at 2493 Hz

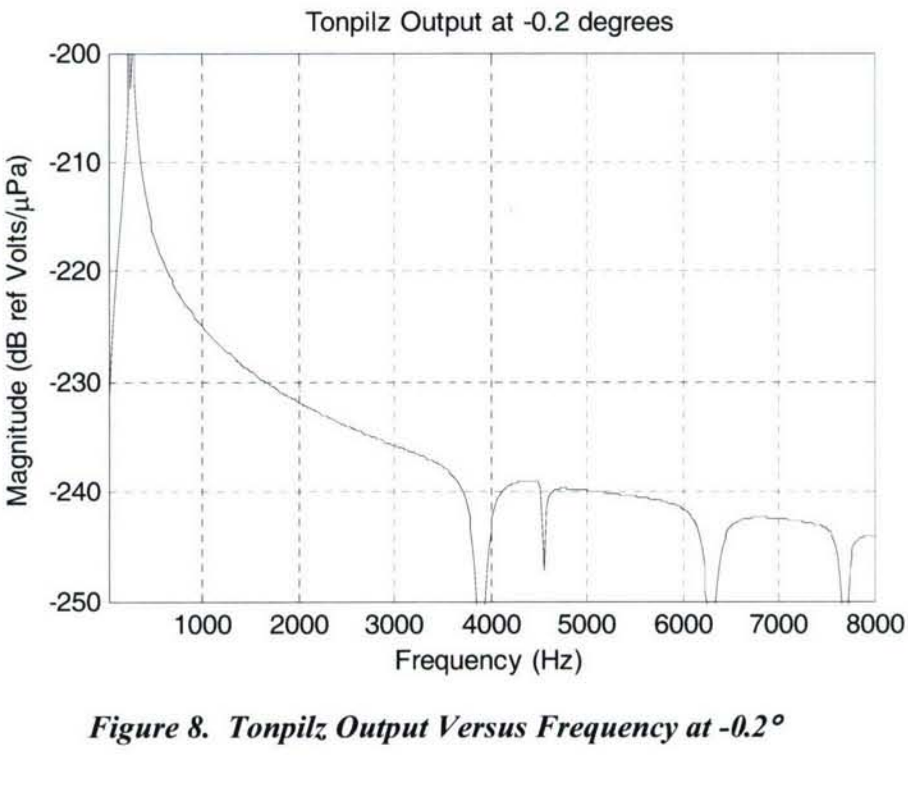

Figure 8. Tonpilz Output Versus Frequency at -0.2°

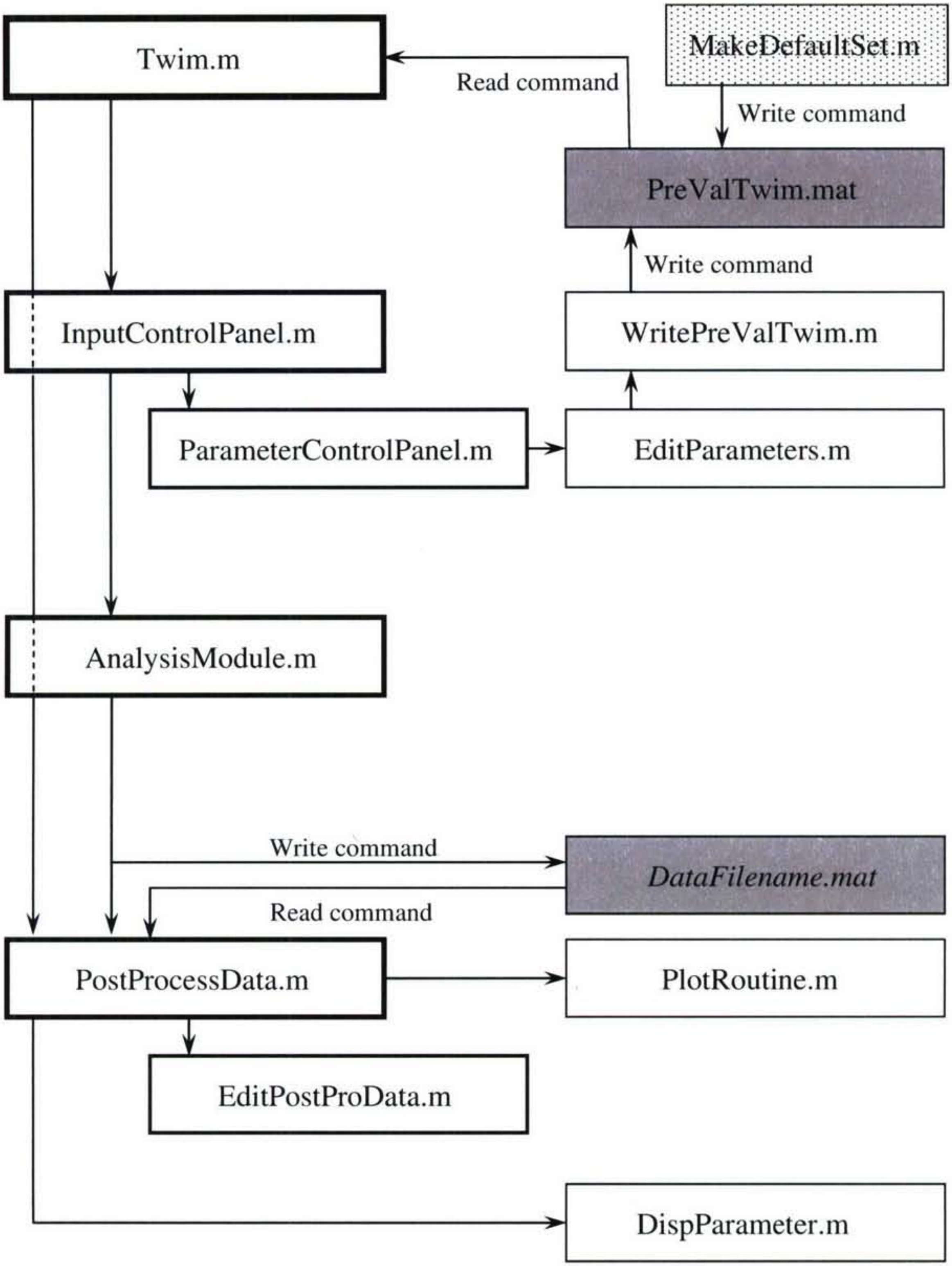

Figure 9. TWIM Program Flowchart

MODEL VALIDATION

The TWIM program is validated by comparison of known analytical solutions for a plate subjected to various loading conditions. The first validation problem is analysis of a thick plate without the Tonpilz transducers. This problem has been solved in closed-form solution in reference 2. The values used to create this example are listed in table 2. Figure 10 is a plot of normal displacement at the bottom of the plate versus wavenumber, and figure 11 is a plot of tangential displacement at the bottom of the plate versus wavenumber. Both figures are at an analysis frequency of 5000 Hz. In both figures, the upper plots are the magnitudes and the lower plots are the phase angles. Additionally, the solid lines are the model output generated with TWIM and the x's are the thick plate theory solutions from reference 2.

Table 2. Parameters Used to Make Verification Problem One

Parameters	Value
Density of Fluid	1025 kg/m^3
Compressional Wavespeed of Fluid	1500 m/s
Young's Modulus of Window	1.00e9 N/m^2
Young's Modulus Loss Factor	0.03 (dimensionless)
Shear Modulus of Window	3.57e8 N/m^2
Shear Modulus Loss Factor	0.03 (dimensionless)
Density of Window	1200 kg/m^3
Thickness of Window	0.100 m
Tonpilz Head Mass	0 kg/m
Tonpilz Tail Mass	0 kg/m
Tonpilz Stiffness	0 N/m^2
Separation Distance of Sensor	0.100 m
Transducer Constant	10 volts m/N
Sensor Window	Off
Sensor Length	0.05 m
Sensor Stack Width	0.01 m
Sensor Stack Height	0.10 m
Minimum Frequency	5000 Hz
Maximum Frequency	5000 Hz
Frequency Points	1
Minimum Wavenumber	0.0 rad/m
Maximum Wavenumber	100 rad/m
Number of Arrival Angle Points	200

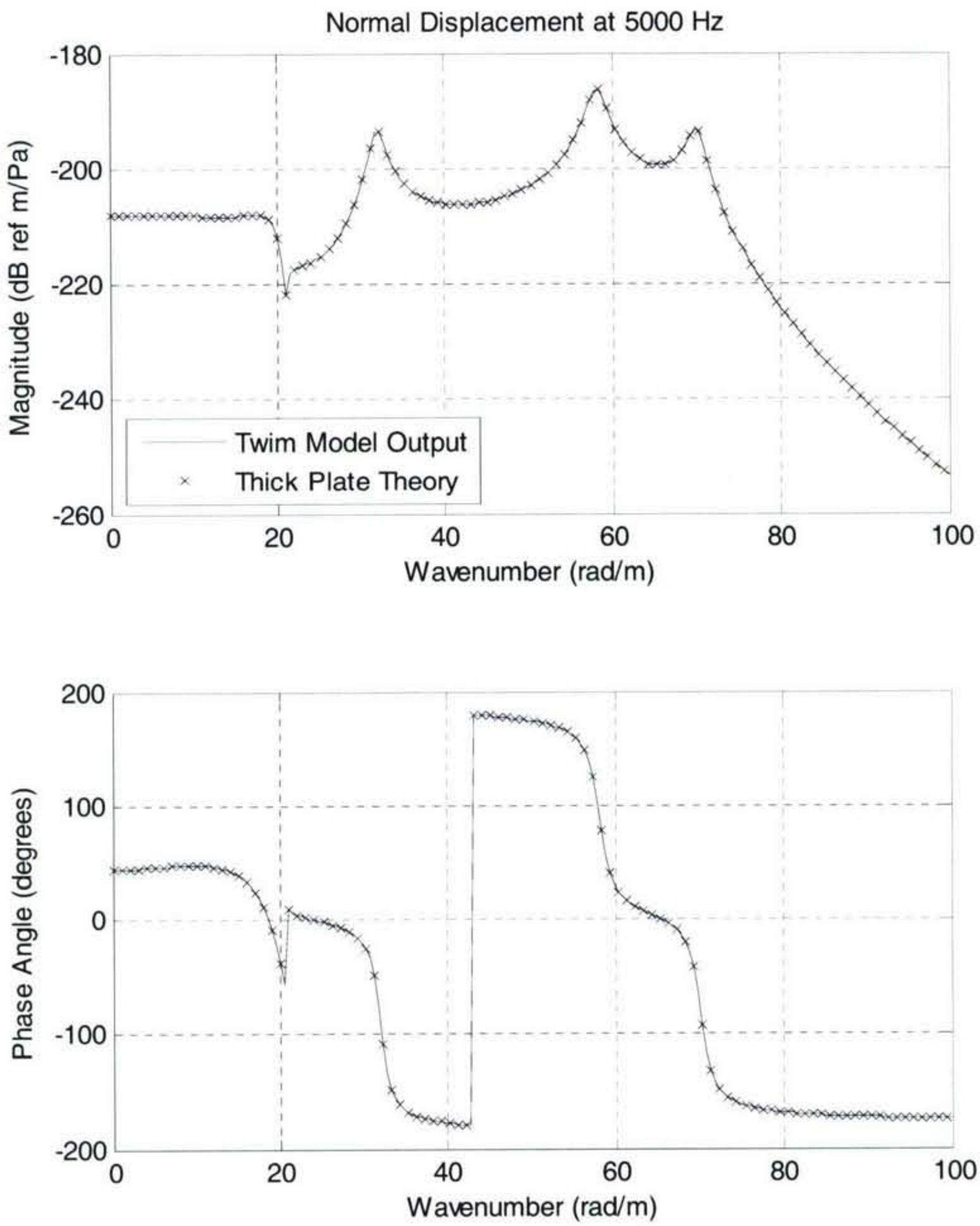

Figure 10. Normal Displacement Versus Wavenumber for Validation Problem One

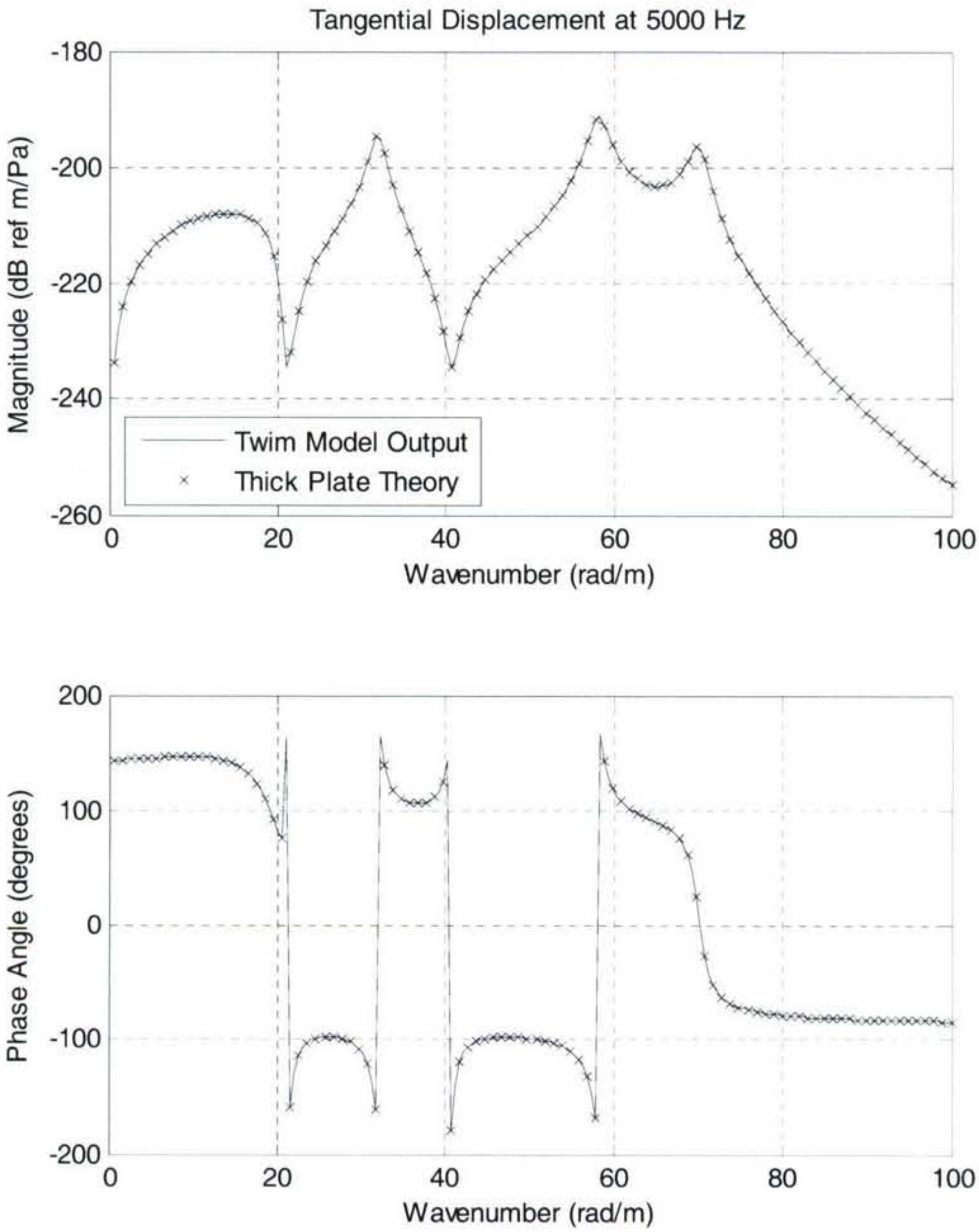

Figure 11. Tangential Displacement Versus Wavenumber for Validation Problem One

The second validation problem is analysis of a nonfluid-loaded thin plate with only the transducer head mass. This problem has been solved in closed-form solution and is discussed in detail in reference 3 (NUWC-NPT TR 11,663). The values used to create this example are listed in table 3. It is noted that to turn the fluid load off in the TWIM program, the density and wavespeed are set to very small values and the displacement values that are output from the program are divided by two. Figure 12 is a plot of normal displacement at the bottom of the plate versus wavenumber, and figure 13 is a plot of tangential displacement at the bottom of the plate versus wavenumber. Both figures are at an analysis frequency of 60 Hz. In figure 12, the solid line is the output from the TWIM program and the x's are the thin plate theory. In figure 13, the solid line is output from the TWIM program. Thin plate theory does not have a tangential degree of freedom; thus, no comparison is present in these plots.

Table 3. Parameters Used to Make Verification Problem Two

Parameters	Value
Density of Fluid	0.1 kg/m^3
Compressional Wavespeed of Fluid	1 m/s
Young's Modulus of Window	7.25e7 N/m^2
Young's Modulus Loss Factor	0.00 (dimensionless)
Shear Modulus of Window	2.50e7 N/m^2
Shear Modulus Loss Factor	0.00 (dimensionless)
Density of Window	1200 kg/m^3
Thickness of Window	0.01 m
Tonpilz Head Mass	1 kg/m
Tonpilz Tail Mass	0 kg/m
Tonpilz Stiffness	0 N/m^2
Separation Distance of Sensor	0.200 m
Transducer Constant	10 volts m/N
Sensor Window	Off
Sensor Length	0.2 m
Sensor Stack Width	0.01 m
Sensor Stack Height	0.10 m
Minimum Frequency	60 Hz
Maximum Frequency	60 Hz
Frequency Points	1
Minimum Wavenumber	0.0 rad/m
Maximum Wavenumber	50 rad/m
Number of Arrival Angle Points	200

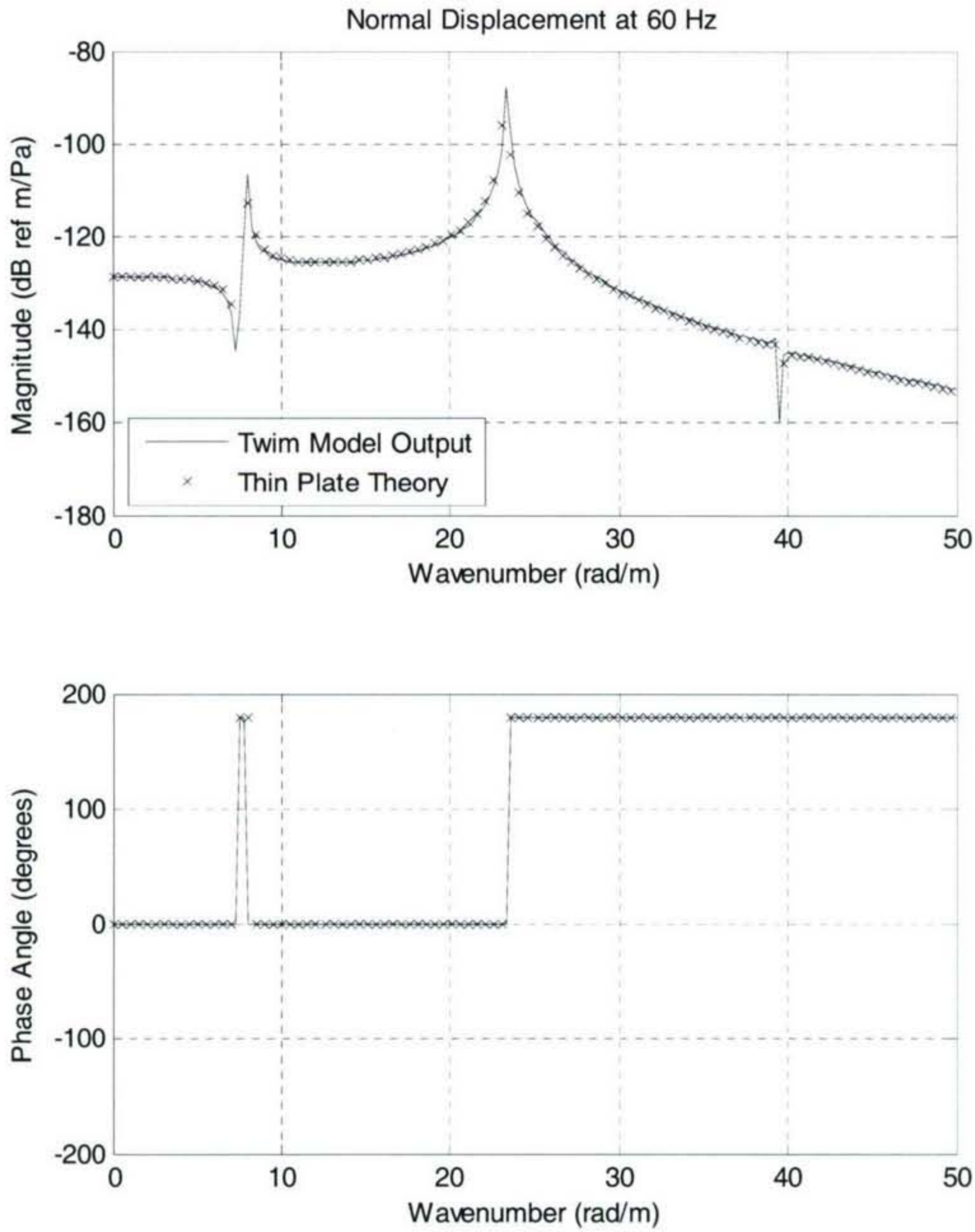

Figure 12. Normal Displacement Versus Wavenumber for Validation Problem Two

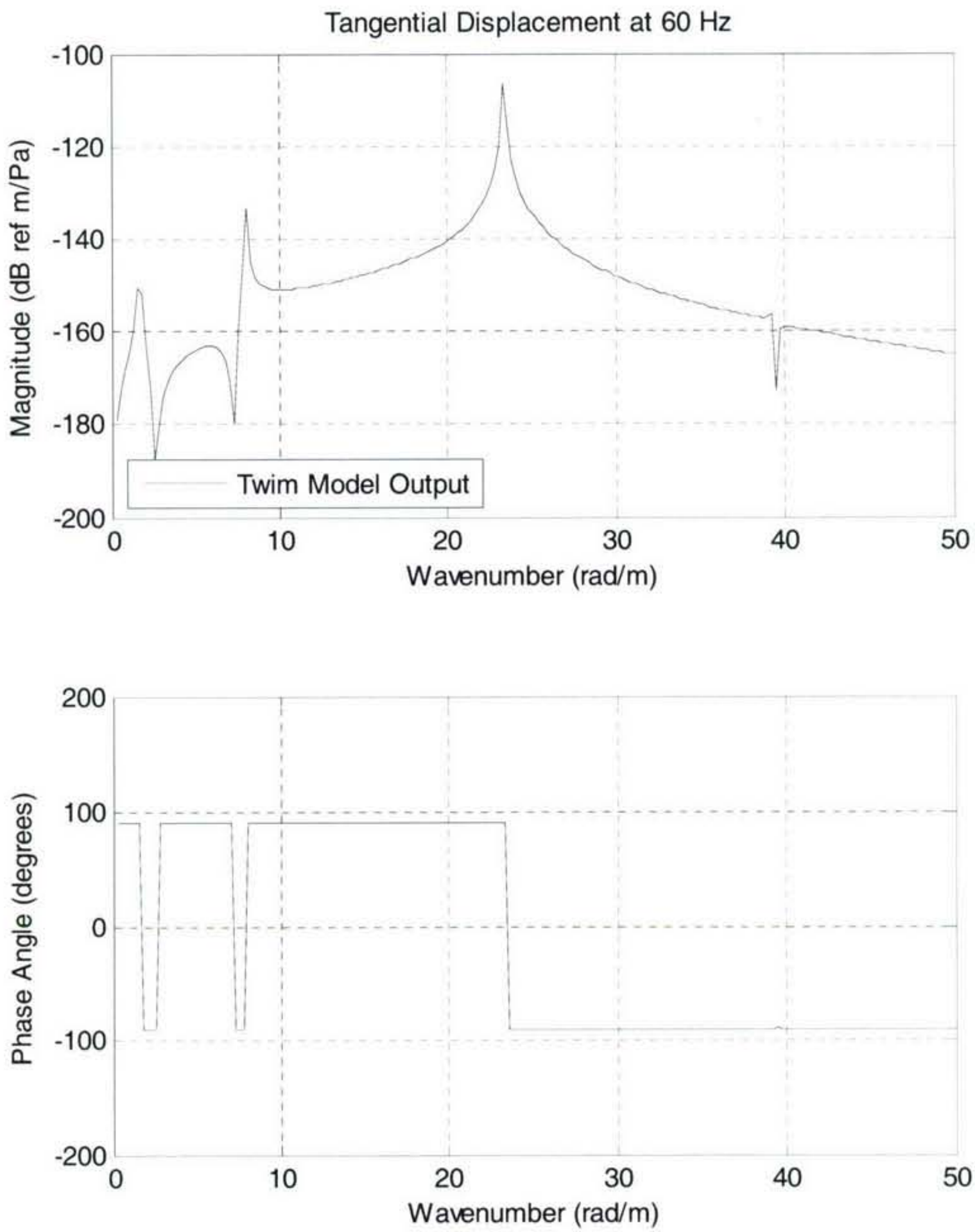

Figure 13. Tangential Displacement Versus Wavenumber for Validation Problem Two

The third validation problem is a nonfluid-loaded thin plate with a Tonpilz transducer array attached to it. The thin plate response in reference 3 can be modified to include a Tonpilz array, and this closed-form solution is used for comparison. The values used to create this example are listed in table 4. Figure 14 is a plot of normal displacement at the bottom of the plate versus frequency, and figure 15 is a plot of the receive voltage output of the transducer versus frequency. Both figures are at an analysis wavenumber of 0 rad/m. Additionally, the solid line is the output from the TWIM program and the x's are the thin plate theory.

Table 4. Parameters Used to Make Verification Problem Three

Parameters	Value
Density of Fluid	1 kg/m^3
Compressional Wavespeed of Fluid	5 m/s
Young's Modulus of Window	7.25e7 N/m^2
Young's Modulus Loss Factor	0.00 (dimensionless)
Shear Modulus of Window	2.50e7 N/m^2
Shear Modulus Loss Factor	0.00 (dimensionless)
Density of Window	1200 kg/m^3
Thickness of Window	0.01 m
Tonpilz Head Mass	0.30 kg/m
Tonpilz Tail Mass	0.25 kg/m
Tonpilz Stiffness	1e7 N/m^2
Separation Distance of Sensor	0.200 m
Transducer Constant	10 volts m/N
Sensor Window	Off
Sensor Length	0.2 m
Sensor Stack Width	0.01 m
Sensor Stack Height	0.10 m
Minimum Frequency	5 Hz
Maximum Frequency	250 Hz
Frequency Points	200
Minimum Wavenumber	0.0 rad/m
Maximum Wavenumber	0.0 rad/m
Number of Arrival Angle Points	1

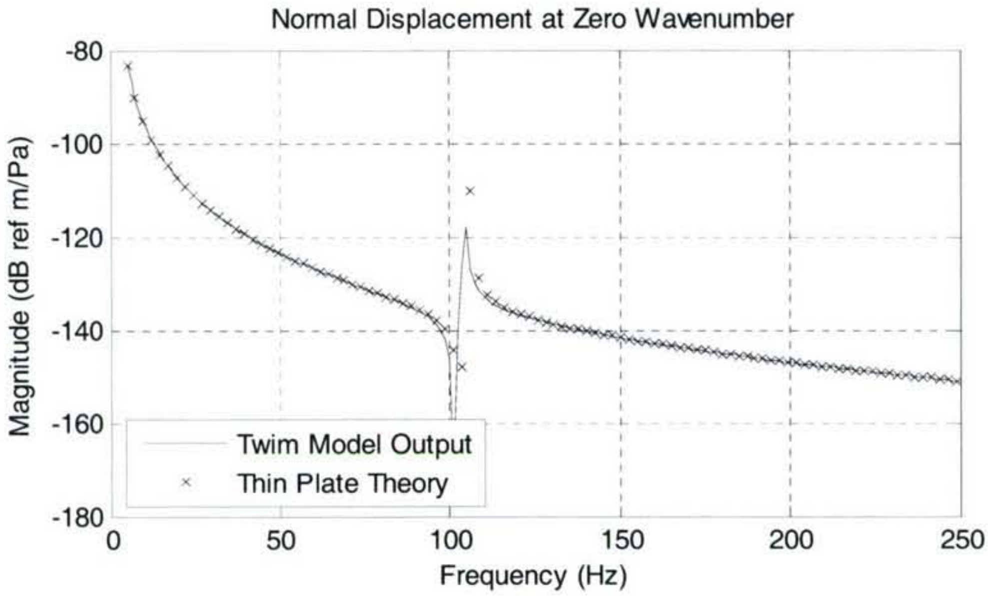

Figure 14. Normal Displacement Versus Frequency for Validation Problem Three

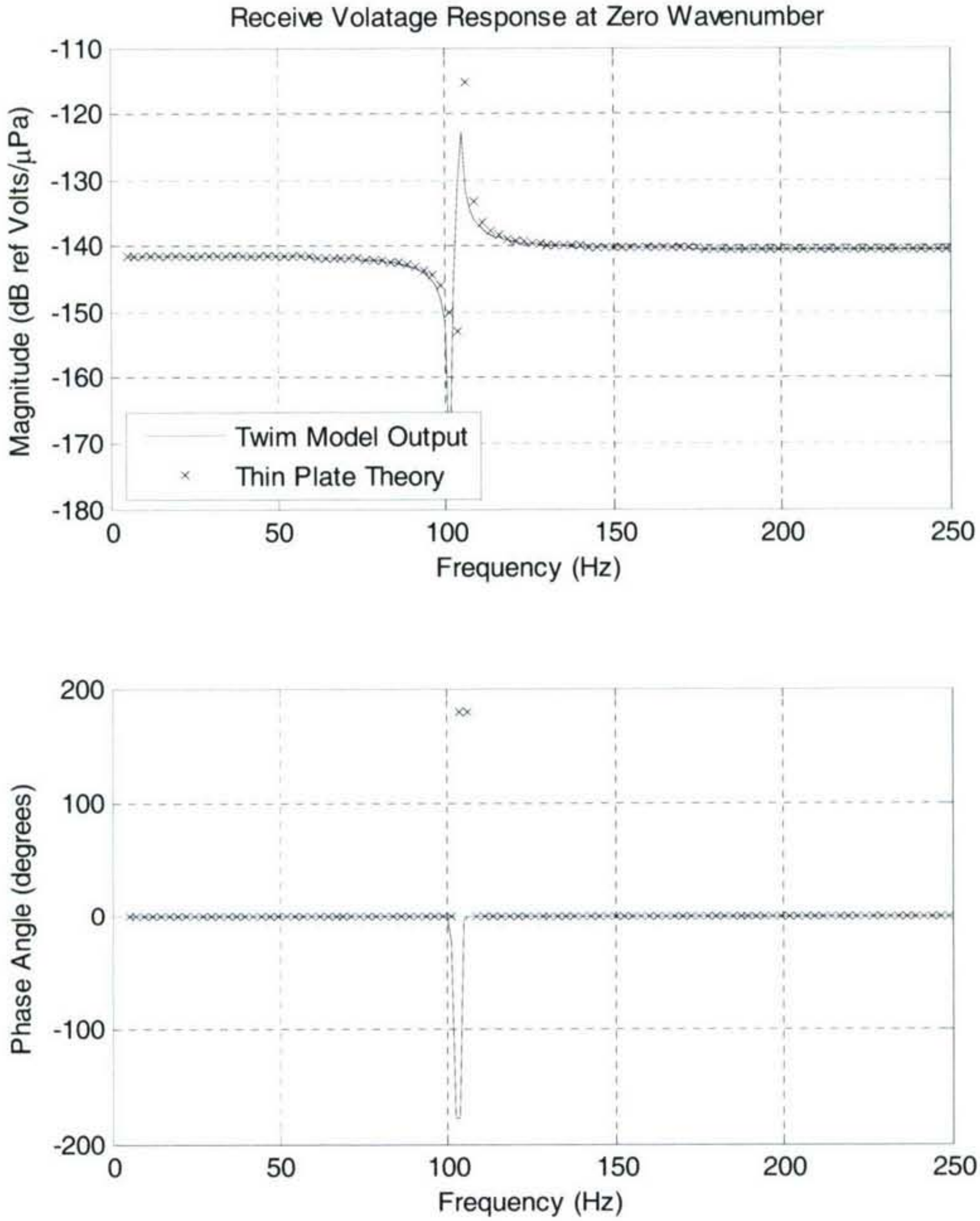

Figure 15. Receive Voltage Response Versus Frequency for Validation Problem Three

The fourth validation problem is analysis of a fluid-loaded thin plate with only the transducer head mass. This problem has been solved in closed-form solution by modifying the thin plate in reference 3 to include a fluid load. The values used to create this example are listed in table 5. Figure 16 is a plot of normal displacement at the bottom of the plate versus wavenumber, and figure 17 is a plot of tangential displacement at the bottom of the plate versus wavenumber. Both figures are at an analysis frequency of 40 Hz. In figure 16, the solid line is the output from the TWIM program and the x's are the thin plate theory. In figure 17, the solid line is output from the TWIM program. Thin plate theory does not have a tangential degree of freedom; thus, no comparison is present in these plots.

Table 5. Parameters Used to Make Verification Problem Four

Parameters	Value
Density of Fluid	1000 kg/m^3
Compressional Wavespeed of Fluid	1500 m/s
Young's Modulus of Window	7.25e7 N/m^2
Young's Modulus Loss Factor	0.00 (dimensionless)
Shear Modulus of Window	2.50e7 N/m^2
Shear Modulus Loss Factor	0.00 (dimensionless)
Density of Window	1200 kg/m^3
Thickness of Window	0.01 m
Tonpilz Head Mass	1.0 kg/m
Tonpilz Tail Mass	0.0 kg/m
Tonpilz Stiffness	0.0 N/m^2
Separation Distance of Sensor	0.200 m
Transducer Constant	10 volts m/N
Sensor Window	Off
Sensor Length	0.2 m
Sensor Stack Width	0.01 m
Sensor Stack Height	0.10 m
Minimum Frequency	40 Hz
Maximum Frequency	40 Hz
Frequency Points	1
Minimum Wavenumber	0.0 rad/m
Maximum Wavenumber	50.0 rad/m
Number of Arrival Angle Points	200

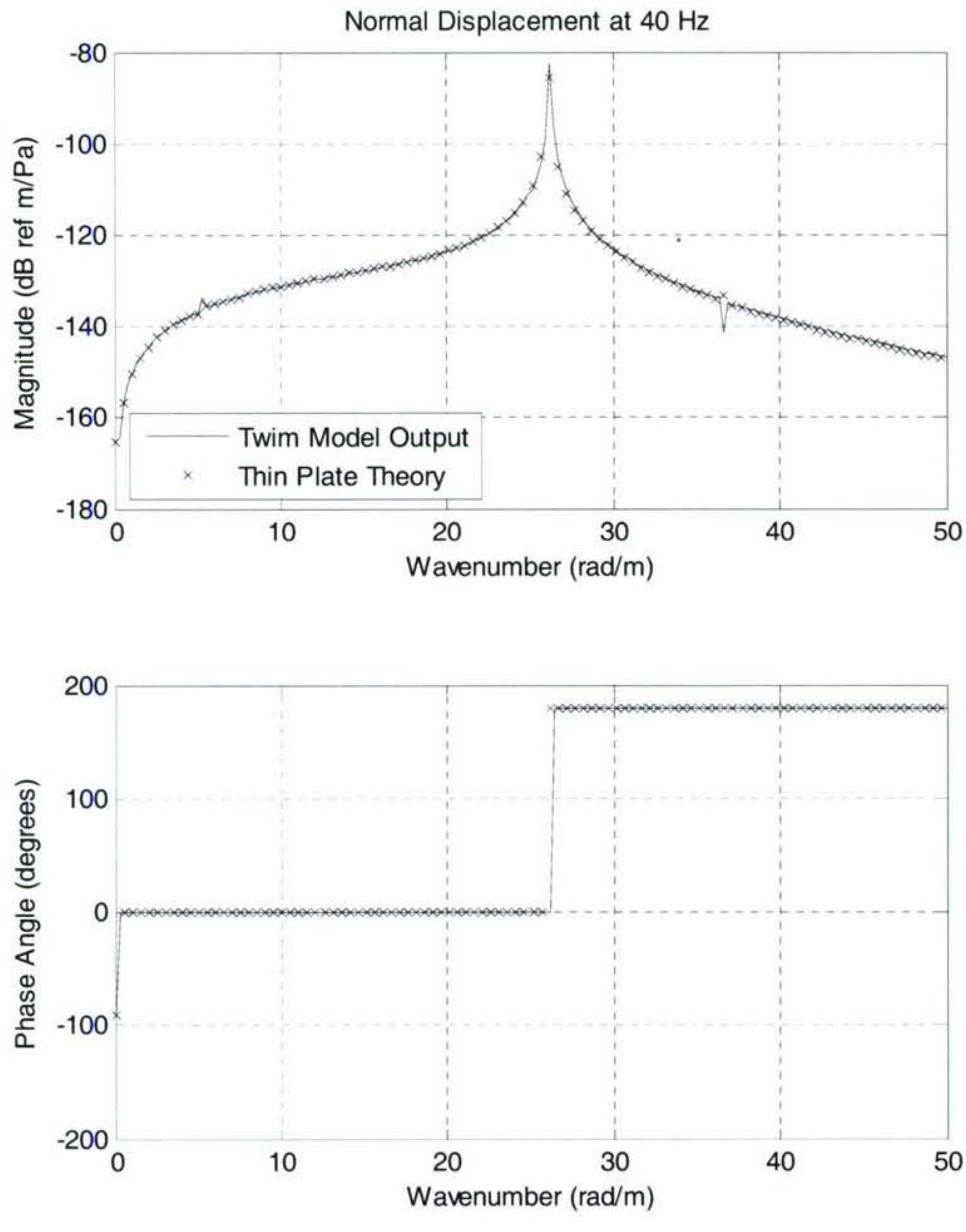

Figure 16. Normal Displacement Versus Wavenumber for Validation Problem Four

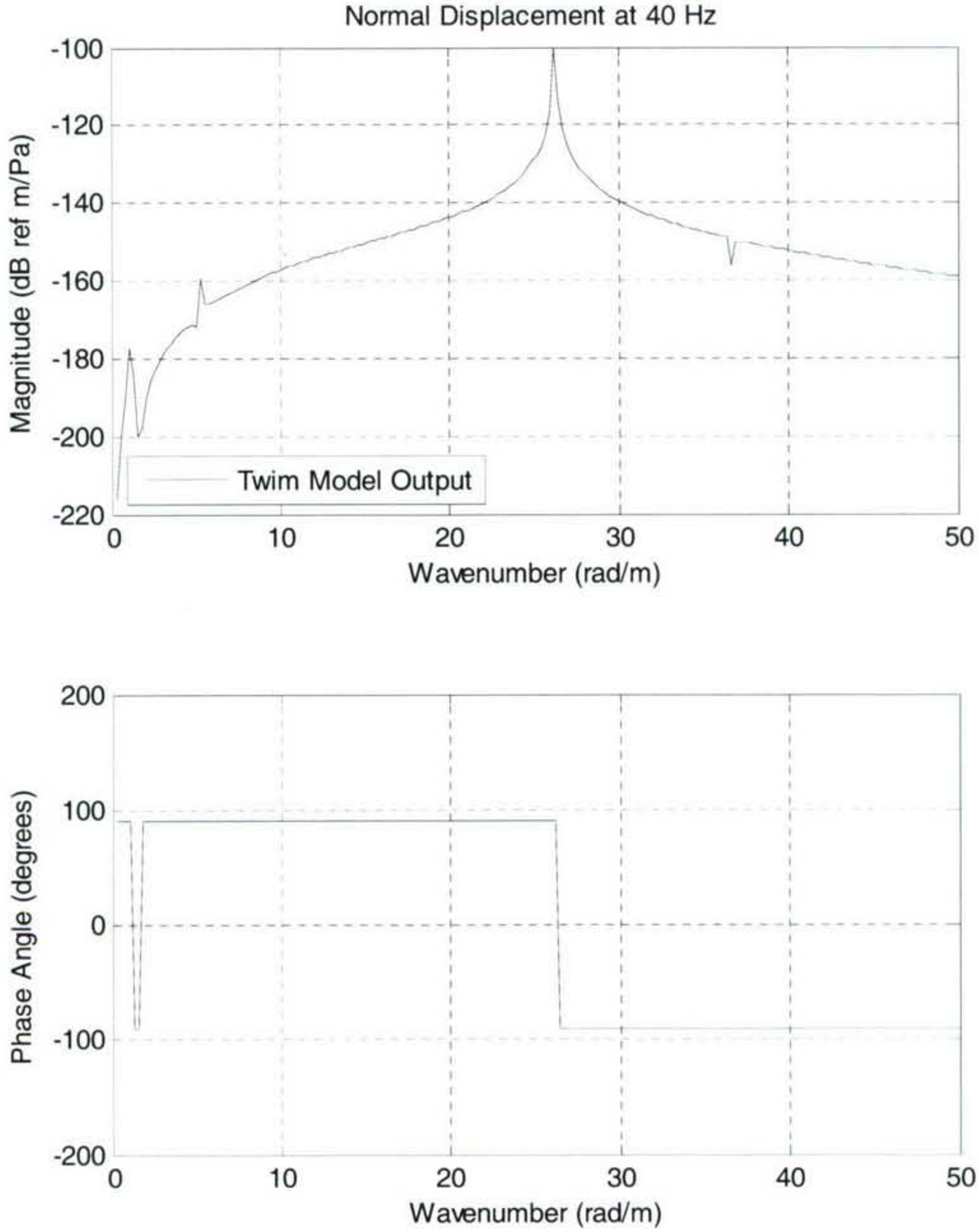

Figure 17. Tangential Displacement Versus Wavenumber for Validation Problem Four

The fifth validation problem is a fluid-loaded thin plate with a Tonpilz transducer array attached to it. The thin plate response in reference 3 can be modified to include a Tonpilz array and the fluid, and this closed-form solution is used for comparison. The values used to create this example are listed in table 6. Figure 18 is a plot of normal displacement at the bottom of the plate versus frequency, and figure 19 is a plot of the receive voltage output of the transducer versus frequency. Both figures are at an analysis wavenumber of 0 rad/m. Additionally, the solid line is the output from the TWIM program and the x's are the thin plate theory.

Table 6. Parameters Used to Make Verification Problem Five

Parameters	Value
Density of Fluid	1000 kg/m^3
Compressional Wavespeed of Fluid	1500 m/s
Young's Modulus of Window	7.25e7 N/m^2
Young's Modulus Loss Factor	0.00 (dimensionless)
Shear Modulus of Window	2.50e7 N/m^2
Shear Modulus Loss Factor	0.00 (dimensionless)
Density of Window	1200 kg/m^3
Thickness of Window	0.01 m
Tonpilz Head Mass	0.3 kg/m
Tonpilz Tail Mass	0.25 kg/m
Tonpilz Stiffness	1e7 N/m^2
Separation Distance of Sensor	0.200 m
Transducer Constant	10 volts m/N
Sensor Window	Off
Sensor Length	0.2 m
Sensor Stack Width	0.01 m
Sensor Stack Height	0.10 m
Minimum Frequency	5 Hz
Maximum Frequency	250 Hz
Frequency Points	200
Minimum Wavenumber	0.0 rad/m
Maximum Wavenumber	0.0 rad/m
Number of Arrival Angle Points	1

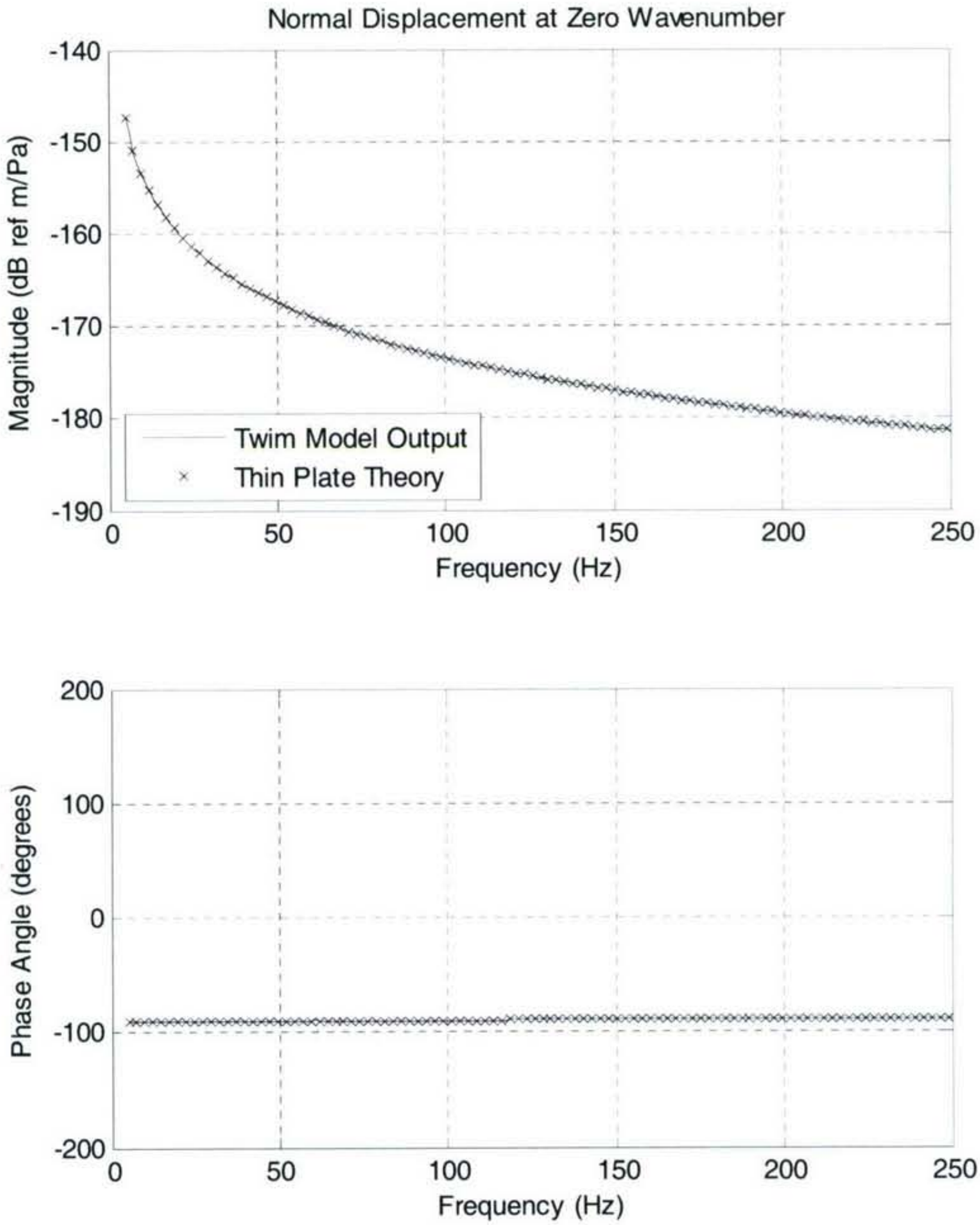

Figure 18. Normal Displacement Versus Wavenumber for Validation Problem Five

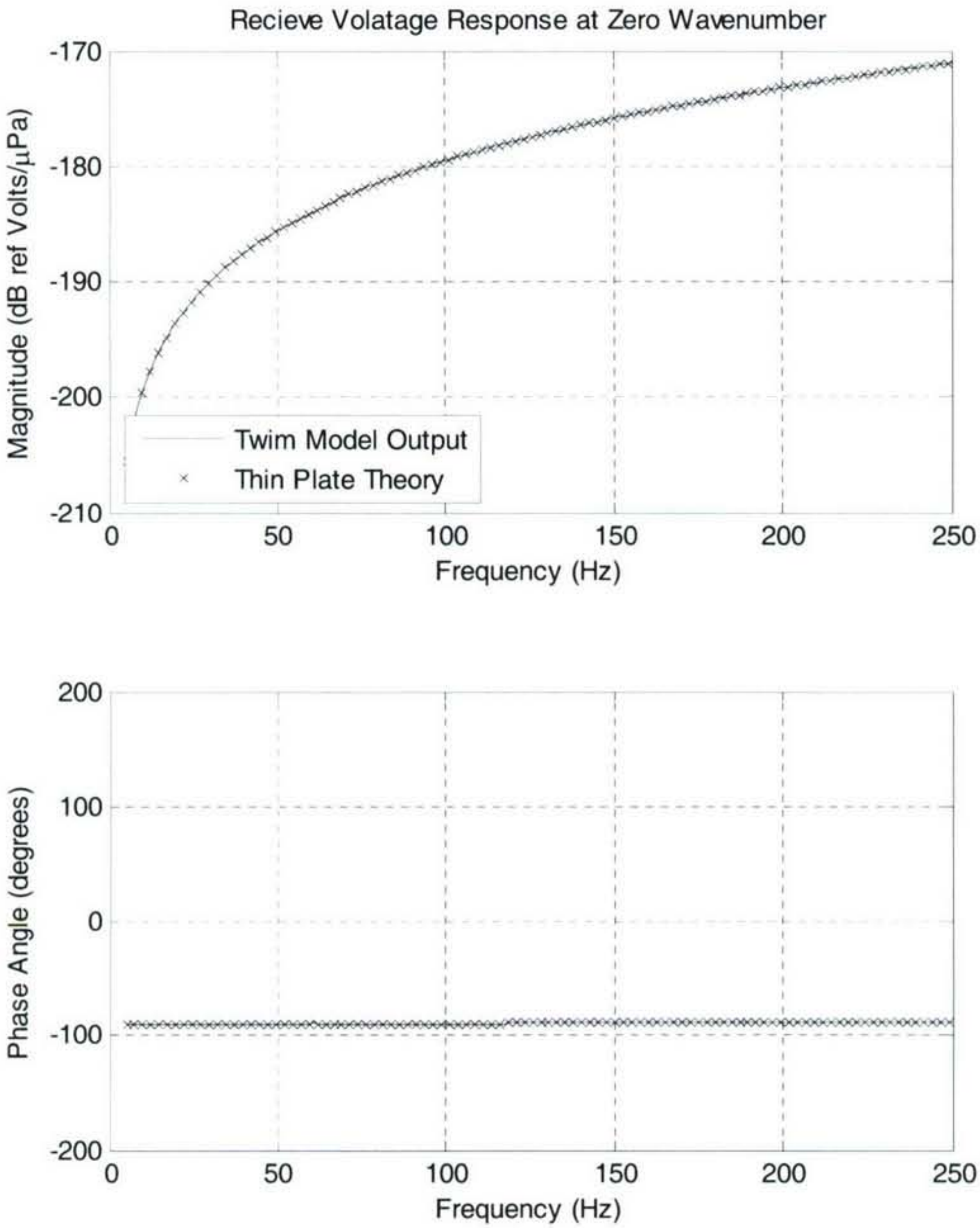

Figure 19. Receive Voltage Response Versus Frequency for Validation Problem Five

The sixth validation problem is included to illustrate the window function in the TWIM program. In this problem, unity value to the window displacement is set at the output of the TWIM program. The window function is then applied to this displacement field, and the result is compared to the theoretical window function. Figure 20 is the resulting plot at 200 Hz. The solid line is the output from the TWIM program and the x's are the theoretical window function. The upper plot is the magnitude and the lower plot is the phase angle. This comparison shows that the effects of the window function in the model are correct.

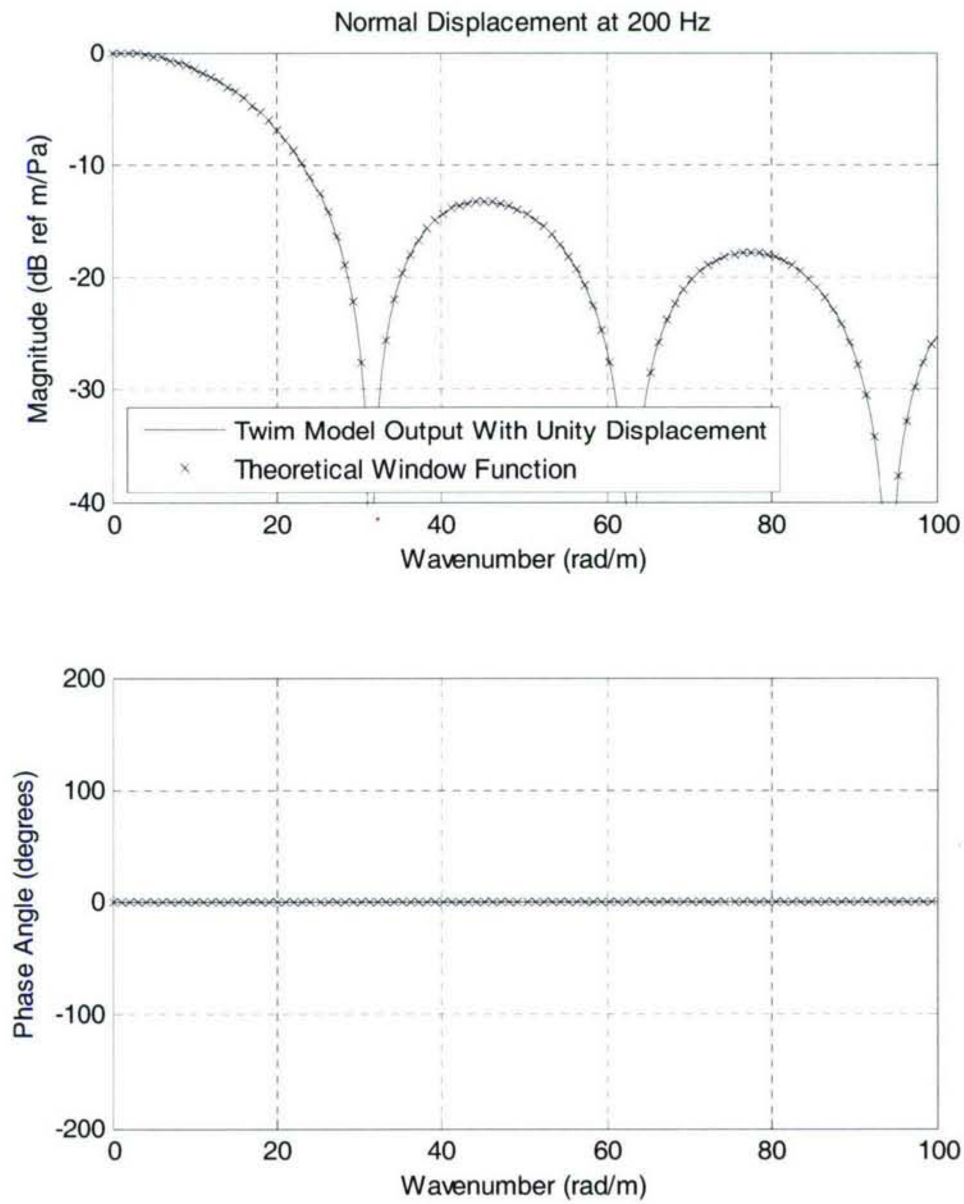

Figure 20. Normal Displacement Versus Wavenumber with Effects of Window Function

REFERENCES

1. Andrew J. Hull, "Dynamic Response of an Insonified Sonar Window Interacting with a Tonpilz Transducer Array," NUWC-NPT Technical Report 11,781, Naval Undersea Warfare Center Division, Newport, RI, 3 January 2007 (UNCLASSIFIED).

2. Andrew J. Hull, "Analysis of a Fluid-Loaded Thick Plate," *Journal of Sound and Vibration*, vol. 279, July 2004, pp. 497-507.

3. Andrew J. Hull, "Dynamic Response of an Elastic Plate Containing Periodic Masses," NUWC-NPT Technical Report 11,663, Naval Undersea Warfare Center Division, Newport, RI, 31 March 2005 (UNCLASSIFIED).

APPENDIX—MATLAB CODE OF ANALYSIS MODULE

```
%--AnalysisModule--is the routine that performs
%      the analysis of the TWIM program
%
%--This program was written by Andrew J. Hull, Code 8212,
%    Autonomous Systems and Technology Department,
%    Naval Undersea Warfare Center Division, Newport,
%    Rhode Island, 02841, (401) 832-5189 in the Fall
%    of 2006.
%
clear all;
close all;
load PreValTwim.mat -mat;
%
FreqVec = linspace ( MinFreq, MaxFreq, FreqPts );
%
if ( Switch_Wavenumber == 1 )
    kVec = linspace ( Mink, Maxk, kPts );
else
    Maxtheta = Maxtheta - (Maxtheta== 90)*0.01;  %--Don't let the value equal  90
    Mintheta = Mintheta + (Mintheta==-90)*0.01;  %--Don't let the value equal -90
    thetaVec = linspace ( Mintheta, Maxtheta, kPts );
end
%
lambda1 = (  2*(G1*(1-(i*G1loss)))^2 - (E1*(1-(i*E1loss)))*(G1*(1-(i*G1loss)))  ) /
...
           (  (E1*(1-(i*E1loss))) - 3*(G1*(1-(i*G1loss)))  );
mu1 = G1*(1-(i*G1loss));
cd1 = sqrt (  ( lambda1 + 2*mu1 ) / ro1 );
cs1 = sqrt (  mu1 / ro1 );
%
bloc = 0.0;
aloc = -h1;
%
%--Let the user write the analysis and output values to a matlab binary file
[datafile,directory] = uiputfile('*.mat', 'Enter Output Data Filename');
OutputFilename = [directory,datafile];
%
%--Open a graphics panel so that the user can see how far along
%     the analysis has progressed
%--Position of GUI box
BoxTitle = ['TWIM version ',TwimVersion];
GUIBoxPosition = 95.0 * [EdgeOffsetInches EdgeOffsetInches WidthInches HeightInches];
GUIBox = figure('NumberTitle','off','Name',BoxTitle,...
                'Position',GUIBoxPosition,'Units','inches');
%
%--Turn the axes off
AxisOff = axes('Visible','off','Units','inches','Position',[0.154*WidthInches
0.444*HeightInches ...
                                                         0.692*WidthInches
0.111*HeightInches]);
%
%--Analysis progression box
XBox(1) = 0.0;XBox(2) = 0.0;YBox(1) = 0.0;YBox(2) = 0.0;
%%ProgressionBox = plot(XBox,YBox,'Visible','off');
ProgressionBox = plot(XBox,YBox);
```

```
set (gca,'YTick',[]);
set (gca,'Xlim',[0 1]);
set (gca,'Ylim',[0 1]);
OldXValue = 0.0;
%
TextVector(1,1) = text('Units','inches',...
                       'Position',[0.354*WidthInches 0.333*HeightInches],...
                       'HorizontalAlignment','center',...
                       'VerticalAlignment','baseline',...
                       'FontSize',TextSize+6,...
                       'String','Analysis Module');
%
TextVector(1,2) = text('Units','inches',...
                       'Position',[0.354*WidthInches 0.278*HeightInches],...
                       'HorizontalAlignment','center',...
                       'VerticalAlignment','baseline',...
                       'FontSize',TextSize+4,...
                       'String','Tonpilz-Window Interaction Model');
%
TextVector(1,3) = text('Units','inches',...
                       'Position',[0.354*WidthInches 0.139*HeightInches],...
                       'HorizontalAlignment','center',...
                       'VerticalAlignment','baseline',...
                       'FontSize',TextSize+4,...
                       'Color','green',...
                       'String','Analysis Progression');
%
%--Put in an Exit option for the user
BoxPosition = [0.346*WidthInches 0.222*HeightInches 0.308*WidthInches
0.0556*HeightInches];
ExitModule = uicontrol('Style','Pushbutton',...
                       'String','Quit Analysis',...
                       'Units','inches',...
                       'BackGroundColor','red',...
                       'Position',BoxPosition,...
                       'FontSize',TextSize,...
                       'Callback','Twim');
%
hold on;
drawnow;
%
%--Number of modes (must be odd)
Nmodes = 11;
nvec = linspace ( -floor(Nmodes/2), floor(Nmodes/2), Nmodes );
%
%--Matrix preallocation for speed
Ux = zeros(kPts,FreqPts);
Uz = zeros(kPts,FreqPts);
Tv = zeros(kPts,FreqPts);
Urow = zeros(4,Nmodes*4);
%
%--The loading vector
Fn(1,1) = -2.0;
Fn(4,1) =  0.0;
zeros41 = zeros(4,1);
Fmatrix = [ ];
for j = 1 : 1 : floor(Nmodes/2)
   Fmatrix = [ Fmatrix; zeros41 ];
end
Fmatrix = [ Fmatrix; Fn ];
for j = 2+floor(Nmodes/2) : 1 : Nmodes
   Fmatrix = [ Fmatrix; zeros41 ];
end
```

```
%
%--Outer Loop in Frequency
for m = 1 : 1 : FreqPts
%
%--Feed the meter
   NewXValue = m / FreqPts;
   Xvector = [OldXValue NewXValue NewXValue OldXValue];
   Yvector = [0 0 1 1];
   fill(Xvector,Yvector,'green','EdgeColor','green','EraseMode','background');
   drawnow
   OldXValue = NewXValue;
%
   w = 2 * pi * FreqVec(m);
   kf = w / cf;
   if ( Switch_Wavenumber == 0 )
      kVec = kf * sin ( thetaVec * ( pi / 180 ));
   end
%
   if ( MT == 0  ||  KS == 0 )
      T = 1;
   else
      T = (-1) * ( ((-w^2)*MT) / (KS-((w^2)*MT)) );
   end
%
   for n = 1 : 1 : kPts
%
      kx = kVec(n);
      kd1 = w / cd1;
      ks1 = w / cs1;
%
      for nindex = 1 : 1 : Nmodes
%
         smalln = nvec(nindex);
         kn = kx + (2*pi*smalln/L);
         alphan1 = sqrt ( kd1^2 - kn^2 );
         betan1 = sqrt ( ks1^2 - kn^2 );
         gammafn = sqrt ( kf^2 - kn^2 );
%
%---------------------------------------------------------------
%
         term1 = ( alphan1*(w^2)*rof ) / gammafn;
         An(1,1) = ( -((alphan1^2)*lambda1) - ((alphan1^2)*2*mu1) - (lambda1*(kn^2)) +
term1 );
         An(1,2) = ( -((alphan1^2)*lambda1) - ((alphan1^2)*2*mu1) - (lambda1*(kn^2)) -
term1 );
         term2 = ( kn*(w^2)*rof ) / gammafn;
         An(1,3) = ( -kn*betan1*2*mu1 + term2 );
         An(1,4) = (  kn*betan1*2*mu1 + term2 );
%
         An(2,1) = ( -2*mu1*kn*alphan1 );
         An(2,2) = (  2*mu1*kn*alphan1 );
         An(2,3) = ( (mu1*(betan1^2)) - (mu1*(kn^2)) );
         An(2,4) = ( (mu1*(betan1^2)) - (mu1*(kn^2)) );
%
         An(3,1) = ( -((alphan1^2)*lambda1) - ((alphan1^2)*2*mu1) - (lambda1*(kn^2)) )
* exp( i*alphan1*aloc);
         An(3,2) = ( -((alphan1^2)*lambda1) - ((alphan1^2)*2*mu1) - (lambda1*(kn^2)) )
* exp(-i*alphan1*aloc);
         An(3,3) = ( -kn*betan1*2*mu1 ) * exp( i*betan1*aloc);
         An(3,4) = (  kn*betan1*2*mu1 ) * exp(-i*betan1*aloc);
%
         An(4,1) = ( -2*mu1*kn*alphan1 ) * exp( i*alphan1*aloc);
         An(4,2) = (  2*mu1*kn*alphan1 ) * exp(-i*alphan1*aloc);
```

```
         An(4,3) = ( (mu1*(betan1^2)) - (mu1*(kn^2)) ) * exp( i*betan1*aloc);
         An(4,4) = ( (mu1*(betan1^2)) - (mu1*(kn^2)) ) * exp(-i*betan1*aloc);
%
         if ( nindex == 1 )
            Amatrix = An;
         else
            [irow icol] = size(Amatrix);
            clear Zeroirowby4;
            Zeroirowby4 = zeros(irow,4);
            Amatrix = [ Amatrix Zeroirowby4 ; Zeroirowby4' An ];
         end
%
%---------------------------------------------------------------
%
         TonpilzLoadX = ( (-w^2) * MH ) / L;

         if ( MT == 0  ||  KS == 0 )
            TonpilzLoadZ = ( (-w^2) * MH ) / L;
         else
            TonpilzLoadZ = (  ( (w^4*MH*MT) - (w^2*KS*(MT+MH)) ) / ( KS - w^2*MT )  )
/ L;
         end
%
         Un(3,1) = TonpilzLoadZ *   i *alphan1 * exp( i*alphan1*aloc);
         Un(3,2) = TonpilzLoadZ * (-i)*alphan1 * exp(-i*alphan1*aloc);
         Un(3,3) = TonpilzLoadZ *   i*kn * exp( i*betan1*aloc);
         Un(3,4) = TonpilzLoadZ *   i*kn * exp(-i*betan1*aloc);
%
         Un(4,1) = TonpilzLoadX *   i*kn * exp( i*alphan1*aloc);
         Un(4,2) = TonpilzLoadX *   i*kn * exp(-i*alphan1*aloc);
         Un(4,3) = TonpilzLoadX * (-i)*betan1 * exp( i*betan1*aloc);
         Un(4,4) = TonpilzLoadX *   i *betan1 * exp(-i*betan1*aloc);
%
         Urow(1:4,(nindex*4)-3:(nindex*4)) = Un(1:4,1:4);
%
%---------------------------------------------------------------
%
      end
%
%
      for nindex = 1 : 1 : Nmodes
         if ( nindex == 1 )
            Umatrix = Urow;
         else
            Umatrix = [ Umatrix; Urow ];
         end
      end
%
%---------------------------------------------------------------
%
      warning('off','all')
      X = inv(Amatrix-Umatrix) * Fmatrix;
      warning('on','all')
%
      startindex = 4*ceil(Nmodes/2) - 3;
%
      zloc = aloc;  %--Location at bottom of plate
      alpha1 = sqrt ( kd1^2 - kx^2 );
      beta1 = sqrt ( ks1^2 - kx^2 );
%
%--At Bottom of plate, stress are zero
%---This part of the code is retained in case evaluation location is changed
%     and the stresses are needed.
```

```
%
%       Szz(n,m) = ( -((alpha1^2)*lambda1) - ((alpha1^2)*2*mu1) - (lambda1*(kx^2)) ) *
X(startindex  ,1) * exp( i*alpha1*zloc) + ...
%                  ( -((alpha1^2)*lambda1) - ((alpha1^2)*2*mu1) - (lambda1*(kx^2)) ) *
X(startindex+1,1) * exp(-i*alpha1*zloc) + ...
%                  ( -2*mu1*kx*beta1 ) * X(startindex+2,1) * exp( i*beta1*zloc) + ...
%                  (  2*mu1*kx*beta1 ) * X(startindex+3,1) * exp(-i*beta1*zloc);
%
%       Szx(n,m) = ( -2*mu1*kx*alpha1 ) * X(startindex  ,1) * exp( i*alpha1*zloc) +
...
%                  (  2*mu1*kx*alpha1 ) * X(startindex+1,1) * exp(-i*alpha1*zloc) +
...
%                  ( (mu1*(beta1^2)) - (mu1*(kx^2)) ) * X(startindex+2,1) * exp(
i*beta1*zloc) + ...
%                  ( (mu1*(beta1^2)) - (mu1*(kx^2)) ) * X(startindex+3,1) * exp(-
i*beta1*zloc);
%
      Uz(n,m) = ( (i*alpha1) * ( X(startindex  ,1)*exp(i*alpha1*zloc) -
X(startindex+1,1)*exp(-i*alpha1*zloc) ) ) + ...
                ( (i*kx) * ( X(startindex+2,1)*exp(i*beta1*zloc) +
X(startindex+3,1)*exp(-i*beta1*zloc) ) );
%
      Ux(n,m) = ( (i*kx) * ( X(startindex  ,1)*exp(i*alpha1*zloc) +
X(startindex+1,1)*exp(-i*alpha1*zloc) ) ) + ...
                ( (i*beta1) * ( -X(startindex+2,1)*exp(i*beta1*zloc) +
X(startindex+3,1)*exp(-i*beta1*zloc) ) );
%
      Tv(n,m) = T * Uz(n,m) * StackHeight * ( KS / SensorLength ) * CAL;
%
   end
%
   if ( Switch_Window == 1 )
      arg = kVec * SensorLength / 2;
      arg = arg + (arg==0)*eps;
      Window = abs ( sin(arg) ./ arg );
%        Szz(:,m) = Window' .* Szz(:,m);
%        Szx(:,m) = Window' .* Szx(:,m);
      Uz(:,m)  = Window' .* Uz(:,m);
      Ux(:,m)  = Window' .* Ux(:,m);
      Tv(:,m)  = Window' .* Tv(:,m);
   end
%
end
%--Convert Output Tonpliz from Volts/Pa to Volts/uPa
Tv = Tv * ( 10^(-6) );
%
clear Amatrix Umatrix X;
%--Runnumber increment
RunNumber = RunNumber + 1;
%--Date and time
DateNow = clock;
if ( DateNow(5) > 9 )
   str2 = [int2str(DateNow(4)),':',int2str(DateNow(5))];
else
   str2 =  [int2str(DateNow(4)),':0',int2str(DateNow(5))];
end
str3 = [int2str(DateNow(2)),'/',int2str(DateNow(3)),'/',int2str(DateNow(1))];
TimeAndDate = [str2,'    ',str3];
%
%--Write the output (results) file to the disk
VariableList = [' Uz Ux Tv FreqVec kVec ', ...
     'rof cf E1 E1loss G1 G1loss ro1 h1 MH MT KS L CAL ',...
     'Switch_Window SensorLength StackWidth StackHeight ',...
```

```
    'MinFreq MaxFreq FreqPts Switch_Wavenumber Mink Maxk kPts ',...
    'Mintheta Maxtheta RunNumber TimeAndDate'];
%--Mac command
%OutputFilename = [OutputFilename,'.mat'];
%--PC command - no change to output filename
eval (['save ',OutputFilename,' -mat',VariableList]);
%
LastOutputFilename = OutputFilename;
WritePreValTwim;
%
%--Go to the graphics post processing routine
PostProcessData;
%-----------------------------------------------------
%-----------------------------------------------------
```

INITIAL DISTRIBUTION LIST

Addressee	No. of Copies
Office of Naval Research (Code 333 – D. Drumheller)	2
Defense Technical Information Center	2
Center for Naval Analyses	1